NASA Technical Memorandum 4127

FREQ: A Computational Package for Multivariable System Loop-Shaping Procedures

Daniel P. Giesy and Ernest S. Armstrong

SEPTEMBER 1989

NASA

NASA Technical Memorandum 4127

FREQ: A Computational Package for Multivariable System Loop-Shaping Procedures

Daniel P. Giesy
PRC Kentron, Inc.
Aerospace Technologies Division
Hampton, Virginia

Ernest S. Armstrong
Langley Research Center
Hampton, Virginia

NASA

National Aeronautics and
Space Administration
Office of Management
Scientific and Technical
Information Division

1989

Contents

Summary

Many approaches in the field of linear, multivariable, time-invariant systems analysis and controller synthesis employ loop-shaping procedures wherein design parameters are chosen to shape frequency-response singular value plots of selected transfer matrices. This report documents a software package, FREQ, for computing within one unified framework many of the most used multivariable transfer matrices for both continuous and discrete systems. The matrices are evaluated at user-selected frequency-response values, and singular values and vectors are computed. FREQ also tabulates maximum and minimum singular values against frequency. Example computations are presented to demonstrate the use of the FREQ code.

Introduction

Many approaches in the field of linear, time-invariant systems analysis and controller synthesis currently employ loop-shaping procedures wherein design parameters are chosen to shape singular-value-magnitude plots of selected transfer matrices. The linear-quadratic-Gaussian (LQG) formulation, ranging from the LQG/loop-transfer-recovery (LTR) approach (Doyle and Stein 1981; "Multivariable Analysis..." 1981; Ridgely and Banda 1986; Stein and Athans 1987) to direct methods utilizing constrained optimization (Mukhopadhyay 1987; Hefner and Mingori 1987), has evolved into a sophisticated methodology for manipulating singular value (Bode-type) plots to specify such properties as stability margin and performance in multiloop feedback. Robust controller techniques (Dorato 1987) implicitly involve singular value plots to design multivariable compensators in the frequency domain. For linear time-invariant systems in general, the feedback properties of plant disturbance attenuation, stability margins, sensor noise response, and sensitivity to plant and sensor variations are quantitatively related to Bode-magnitude plots of the singular values of the return difference and associated inverse return difference matrices (Safonov, Laub, and Hartmann 1981).

A standard (Doyle and Stein 1981) plant/compensator configuration employed in many of the loop-shaping methodologies is shown in figure 1. The block diagram of figure 1(a) shows a plant with transfer matrix $\mathbf{G}_0$ interconnected in unity-gain feedback fashion with a compensator $\mathbf{K}_0$ defining a plant input u. The system can be forced by exogenous commands r, measurement noise n, and disturbances d. In figure 1(a), all disturbances have been reflected to the measured output y. In the compensator design and system analysis process, a number of important transfer matrices arise. Depending on whether the loop in figure 1(a) is broken at the plant input or plant output, the loop-gain transfer matrix $\mathbf{G}_L$ is either $\mathbf{G}_L = \mathbf{K}_0\mathbf{G}_0$ or $\mathbf{G}_L = \mathbf{G}_0\mathbf{K}_0$, respectively. Figures 1(b) and 1(c) also give

$$y = \mathbf{G}_0\mathbf{K}_0(\mathbf{I} + \mathbf{G}_0\mathbf{K}_0)^{-1}(r - n) + (\mathbf{I} + \mathbf{G}_0\mathbf{K}_0)^{-1}d$$

and

$$r - y = (\mathbf{I} + \mathbf{G}_0\mathbf{K}_0)^{-1}(r - d) + \mathbf{G}_0\mathbf{K}_0(\mathbf{I} + \mathbf{G}_0\mathbf{K}_0)^{-1}n$$

where $\mathbf{I}$ denotes the identity matrix. Alternatively, for a reference input u_r, if r is replaced by $\mathbf{G}_0u_r$,

$$u = \mathbf{K}_0\mathbf{G}_0(\mathbf{I} + \mathbf{K}_0\mathbf{G}_0)^{-1}(u_r - \tilde{d} - \tilde{n})$$

and

$$u_r - u = (\mathbf{I} + \mathbf{K}_0\mathbf{G}_0)^{-1}u_r - \mathbf{K}_0\mathbf{G}_0(\mathbf{I} + \mathbf{K}_0\mathbf{G}_0)^{-1}(\tilde{d} + \tilde{n})$$

where $\tilde{d}$ and $\tilde{n}$ are defined by the relations

$$\mathbf{G}_0\tilde{d} = d$$

and

$$\mathbf{G}_0\tilde{n} = n$$

It has also been shown (Doyle and Stein 1981) that the stability robustness to unstructured uncertainties at the plant output can be measured in terms of inequalities involving singular values of

$$\mathbf{I} + (\mathbf{G}_0\mathbf{K}_0)^{-1}$$

Tolerances for unstructured uncertainties at the plant input involve singular values of

$$\mathbf{I} + (\mathbf{K}_0\mathbf{G}_0)^{-1}$$

For structured uncertainties at the plant input or output, structured singular value methodology (Doyle, Wall, and Stein 1982) should be employed. Loop-shaping techniques based on singular value tolerances work well for unstructured uncertainties, but they are likely to give overly conservative designs in the structured uncertainty case. However, singular value methodology is computationally easier to implement and can be used to give upper bounds on structured singular values.

Thus, for a given $\mathbf{G}_L$, the following arise:

The closed-loop response (complementary sensitivity) matrix	$\mathbf{G}_L(\mathbf{I} + \mathbf{G}_L)^{-1}$
The return difference matrix	$(\mathbf{I} + \mathbf{G}_L)$
The inverse return difference matrix	$(\mathbf{I} + \mathbf{G}_L^{-1})$
The sensitivity matrix	$(\mathbf{I} + \mathbf{G}_L)^{-1}$

Various loop-shaping, singular-value-based analysis and design methodologies employing these transfer matrices can be found in the previously cited references. The purpose of this report is to address computational issues common to all loop-shaping, singular-value-based methodologies—i.e., the numerical evaluation of the transfer matrices, the computation of their singular value bounds, and the tabulation of data for loop-shaping plots.

This report documents a software package, FREQ, for computing within one unified framework a variety of frequency-response transfer matrices including the aforementioned plant, compensator, loop gain(s), closed-loop response, return difference, inverse return difference, and sensitivity matrices. For the case in which the plant $\mathbf{G}_0$ and/or compensator $\mathbf{K}_0$ are constructed through given linear, time-invariant state-space realizations $(\mathbf{A}_g, \mathbf{B}_g, \mathbf{C}_g, \mathbf{D}_g)$ and/or $(\mathbf{A}_k, \mathbf{B}_k, \mathbf{C}_k, \mathbf{D}_k)$, respectively, FREQ internally calculates, for selected values of the frequency-response variable $s = j\omega$, the plant and/or compensator transfer matrix as

$$\mathbf{G}_0(s) = \mathbf{C}_g(s\mathbf{I} - \mathbf{A}_g)^{-1}\mathbf{B}_g + \mathbf{D}_g$$

$$\mathbf{K}_0(s) = \mathbf{C}_k(s\mathbf{I} - \mathbf{A}_k)^{-1}\mathbf{B}_k + \mathbf{D}_k$$

where s represents the Laplace transform parameter, ω represents the frequency, and $j^2 = -1$. For other forms of plant or compensator, the user has the freedom to externally compute and directly input the transfer matrix. Generally, at the option of the user, FREQ provides the capability of computing, for selected values of $s = j\omega$, some or all of the transfer matrices, their singular values and vectors, as well as singular-value-magnitude tabulations of maximum and minimum singular values against frequency ω. FREQ also includes the capability of performing the foregoing family of computations for discrete system transfer matrices in which s is replaced by $e^{j\bar{\omega}}$, where $0 \leq \bar{\omega} \leq \pi$. Specific details of the FREQ package are presented in the sections to follow.

The next section, "Functional Description of Software" (see p. 3), explains in linear algebraic terms the calculations that can be performed by FREQ. The following three sections may be viewed as a FREQ users manual. "Use of Software: Initialization and Output" (see p. 4) describes the function of some internal parameters used by FREQ and explains how to (re)initialize them. "Use of Software: Computational Routines" (see p. 7) explains how to call those FREQ subroutines that comprise the user interface to the working part of the package. This section is addressed to the analysis of transfer matrices arising from continuous systems; but, with a few additional considerations, this section applies also to discrete systems. "Use of Software: Discrete System Analysis" (see p. 17) gives the additional information necessary to use FREQ to analyze transfer matrices arising from discrete systems. "Software Notes" (see p. 17) goes into more detail about how FREQ performs its computations. Information presented in this section helps the user to make the correct choices of program options and to understand any error messages that FREQ might generate. It also explains how to access the code, in which language FREQ is coded, and which third-party code is included. The section entitled "Example" (see p. 21) demonstrates the use of the code.

Five appendixes give information primarily of interest to programmers using the FREQ software package. "Appendix A—Guide to Singular Value Tabulation" (see p. 22) gives an annotated listing of SDTAB, one of the tabulation drivers, which can be used as a template by the user who has special needs in a tabulation driver. "Appendix B—Selected Subroutine Preambles" (see p. 26) gives the program preambles of the FREQ user interface subroutines. In the interest of economy of distribution, these have been left out of the versions of the source code distributed by the NASA-sponsored Computer Software Management and Information Center (COSMIC). "Appendix C—Symbolic Names of Global Entities" (see p. 52) lists the symbolic names of global entities such as subroutines, common blocks, etc., used by FREQ. The user needs to avoid using these as global names in his own code. In addition, these global entities are correlated with the software organization charts in figures 2 and 3. "Appendix D—Example Program Listing" (see p. 53) contains the listing of the program used to produce the example. "Appendix E—Output From Example Computation" (see p. 74) contains the output from executing the example program of appendix D.

Use of the FREQ package is demonstrated in the section "Example" (see p. 21) with data taken from a study in which a fine-pointing attitude control system was designed for a large, flexible hoop/column space antenna (Joshi, Armstrong, and Sundararajan 1986; Sundararajan, Joshi, and Armstrong 1987). The control system was designed to possess stability robustness with respect to high-frequency, unmodeled antenna dynamics, and a prototype version of FREQ was employed to obtain the robustness-barrier design plots. Similar studies and uses of FREQ for other large space antenna configurations may be found in Joshi and Armstrong (1987); Armstrong, Joshi, and Stewart (1988); and Joshi, Rowell, and Armstrong (1988). Each such application of FREQ indicated new user needs and preferences. Since the FREQ package has evolved through a sequence of control analysis and design applications, it is felt that it provides a body of tested software incorporating most of the desired properties and options needed by the practicing engineer.

Functional Description of Software

The primary function of the FREQ package is to provide a frequency-domain analysis tool for the generic plant/compensator configuration shown in figure 1 where both the plant and compensator were described by linear, time-invariant state-space realizations. In addition, FREQ contains other options that have been added at the request of early users.

FREQ operates at two levels. Subroutines SIGCAL and SIGDYN perform the analysis at a single value of the frequency parameter. Subroutines SIGTAB and SDTAB, which accept multiple values of the frequency parameter, perform the analysis at each of these values by making repeated calls to SIGCAL or SIGDYN, respectively.

For each single value of the frequency parameter, FREQ must perform a number of tasks. The first task of FREQ is to generate a transfer matrix $\mathbf{G}_L(s)$ at a value of the Laplace variable $s = j\omega$ (where $j^2 = -1$) determined for a user-specified frequency ω or to generate a z-transform matrix $\mathbf{G}_L(z)$ at a value of the variable $z = e^{j\bar{\omega}}$ determined for a user-specified normalized frequency $\bar{\omega}$. A typical "normalized frequency" is angular frequency multiplied by sampling period. For simplicity of presentation, we will refer only to the continuous case ($s = j\omega$) in the following discussion. The reader interested in the discrete case ($z = e^{j\bar{\omega}}$) can substitute z for s in his own mind and take note of the later section of this paper, "Use of Software: Discrete System Analysis" (see p. 17).

The user has several options at this stage: $\mathbf{G}_L(s)$ may be $\mathbf{G}_0(s), \mathbf{K}_0(s), \mathbf{K}_0(s)^{\pm 1}\ \mathbf{G}_0(s)^{\pm 1}$, or $\mathbf{G}_0(s)^{\pm 1}\ \mathbf{K}_0(s)^{\pm 1}$. The user may pass to FREQ system matrices that constitute a linear, time-invariant state-space realization of $\mathbf{G}_0(s)$ (and/or $\mathbf{K}_0(s)$), and FREQ will calculate $\mathbf{G}_0(s)$ (and/or $\mathbf{K}_0(s)$) internally. Alternatively, the user may calculate the matrix $\mathbf{G}_0(s)$ (and/or $\mathbf{K}_0(s)$, if it is used) externally to FREQ and pass it in. An option is offered to compute the system matrix eigenvalues.

Next, FREQ determines, at the option of the user, a matrix $\mathbf{G}(s)$ that may be either the $\mathbf{G}_L$ just calculated or one of $\mathbf{G}_L(\mathbf{I}+\mathbf{G}_L)^{-1}, (\mathbf{I}+\mathbf{G}_L)^{-1}, (\mathbf{I}+\mathbf{G}_L^{-1})$, or $(\mathbf{I}+\mathbf{G}_L)$. The option exists to compute the singular values of $\mathbf{G}$, and if singular values are computed, a suboption is provided to compute singular vectors.

The subroutines SIGTAB and SDTAB which accept multiple frequency values in an array are offered since one of the intended uses of FREQ is to provide the information needed to plot the maximum and/or minimum singular values of $\mathbf{G}(j\omega)$ or $\mathbf{G}(e^{j\bar{\omega}})$ as a function of ω or $\bar{\omega}$, respectively. Provision is made to perform the preceding calculations for each value of ω or $\bar{\omega}$ in an array (which might be user supplied or internally calculated from user specifications) and to return the corresponding arrays with the maximum and minimum singular values of the chosen $\mathbf{G}$ matrix. Not all "single ω" options are available here. Primarily, $\mathbf{G}_0$ (and $\mathbf{K}_0$, if used) must be internally calculated from linear state-space realizations. (The user for whom this restriction is a burden is invited to use FREQ subroutines SIGTAB or SDTAB as templates for constructing a singular value tabulation driver. To aid in this construction, an annotated copy of SDTAB is given in appendix A.)

FREQ uses the efficient, multivariable frequency-response calculation subroutines developed by Laub (Laub 1981; Lehtomaki 1981; Laub 1986) for computations of the form $\mathbf{C}(s\mathbf{I}-\mathbf{A})^{-1}\mathbf{B}$, where $(\mathbf{A},\mathbf{B},\mathbf{C})$ are constant real matrices of compatible dimension. The representation of the transfer matrix $\mathbf{C}(s\mathbf{I}-\mathbf{A})^{-1}\mathbf{B}$ is modified through a similarity transformation applied to $\mathbf{A}$ to produce the form $\mathbf{C}'(s\mathbf{I}-\mathbf{A}')^{-1}\mathbf{B}'$, where $\mathbf{A}'$ is upper Hessenberg (i.e., a matrix whose elements are zero below the first subdiagonal). More information on this process is included below in the subsection "Plant/Compensator Parameters" (see p. 10) and also in "Software Notes" (see p. 17). This transformation is independent of s and need be done only once even if the transfer matrix is to be evaluated for many values of s. The efficiency arises in that the calculation of $(s\mathbf{I}-\mathbf{A}')^{-1}\mathbf{B}'$ with $\mathbf{A}'$ upper Hessenberg is much faster than that of $(s\mathbf{I}-\mathbf{A})^{-1}\mathbf{B}$ with a general $\mathbf{A}$ matrix (Laub 1981). Laub's software includes a mechanism that will track whether the $\mathbf{A}, \mathbf{B}$, and $\mathbf{C}$ matrices have undergone this transformation.

Use of Software: Initialization and Output

This section and the two sections that follow provide a FREQ users manual and describe the overall structure of the code. This section describes how to initialize FREQ and set certain parameters and what effects these parameters have on output and error processing. The next section describes the functions of and the calling sequences for the five subroutines intended as the user interface into FREQ. The section after that gives the additional information that the user needs to use FREQ for the analysis of discrete systems.

Initialization and Parameter Management

FREQ manages its output and error processing through a number of parameters that are stored in common blocks. Before initializing FREQ, the user must open the input file (if used), the output file, and the error message file (if different from output). Then, the user *must* do at least one of the several actions that will initialize these parameters before calling any of the FREQ computational routines.

Figure 2 shows the structure of initialization and output routines in FREQ. The lines with arrowheads denote data flow, whereas the lines without arrowheads show software-calling hierarchy. The dotted box and lines indicate that any other software available that knows the FREQ initialization protocols may be used.

The parameters set by initialization routines are given as follows:

NIN	the external unit identifier of the input file; the default value is 5.
NOUT	the external unit identifier of the output file; the default value is 6.
NERR	the external unit identifier of the file for error messages may be the same as the output file; the default value is 6.
NLP	the number of lines on a page of output; the default value of 59 is suitable for 11-in-long paper printed at 6 lines per inch.

NCL	the line length (in characters) of an output line, forced by FREQ to be either 80 or 132; the default value of 132 is suitable for the typical long-carriage line printer.
IEROPT	an integer error-option flag; the default value of 0 tells FREQ to stop when a fatal error is detected. An error is considered fatal if FREQ is unable to make requested calculations because of this error (except for certain eigenvalue calculations as explained later).
TITLE	a CHARACTER*80 variable that contains information printed at the top of each output page; default is all blanks.

These parameters can be set (or reset) by calls to any of three routines, FREQNV, FRDTIT, or FREQNT. In addition, FREQNV can also be used to identify the current values of the parameters, and subroutine ORACES returns the current value of IEROPT while setting a new value. These routines are called in the following manner.

FREQNT is called with no arguments:

```
CALL FREQNT
```

This sets all parameters to their default values.

FRDTIT is called with no arguments:

```
CALL FRDTIT
```

This sets all parameters except TITLE to their default values. It then reads the next line of unit 5 (since NIN has just been set to default) and uses it for TITLE. FRDTIT is the analogue of ORACLS (Armstrong 1978a, 1978b, 1980) routine RDTITL.

FREQNV is called with arguments:

```
CALL FREQNV(IOPT,NIN,NOUT,NERR,NLP,NCL,IEROPT,TITLE)
```

IOPT is a user-set option flag that specifies what is to be accomplished by this call to FREQNV.

IOPT = 1:	FREQNV uses the values set by the user in its remaining arguments to initialize the corresponding FREQ parameters. If the user-input value of NCL is less than 132, FREQNV resets NCL to 80. Otherwise, NCL is set to 132.
IOPT = 0:	FREQNV treats all parameters except TITLE exactly as it does in the case that IOPT = 1. TITLE is read from unit NIN which the user has just specified. The new value is returned to the user in calling sequence argument TITLE.
IOPT = −1:	No FREQ parameters are reset; the current values are returned to the user in the arguments. This option should not be exercised until the parameters have been set by some other action.

Since all arguments of FREQNV except IOPT can be changed by FREQNV, safe programming practice suggests calling FREQNV with variables in these positions instead of actual values. The user who wishes mostly default values for the parameters with a few exceptions can use the following technique:

```
      CHARACTER*80 TITLE
      ...
      CALL FREQNT
      CALL FREQNV(-1,NIN,NOUT,NERR,NLP,NCL,IEROPT,TITLE)
C     LINES OF CODE RESETTING THE FEW EXCEPTIONS, E.G.,
      NCL = 80
      NERR = 7
```

```
        TITLE = 'TEST RUN'
        CALL FREQNV(+1,NIN,NOUT,NERR,NLP,NCL,IEROPT,TITLE)
```

The call to FREQNT sets default values, the first call to FREQNV puts these in the user's variables, the next few lines reset the ones that the user wants changed, and the second call to FREQNV informs FREQ of the changes.

<u>ORACES</u> is called with arguments:

```
   CALL ORACES(NEROPT, LEROPT)
```

This subroutine resets the parameter IEROPT to a user-specified value while remembering the value that was in IEROPT before ORACES was called.

NEROPT	On input, the new value to which IEROPT will be set, not changed by subroutine ORACES.
LEROPT	Not used on input; on output, ORACES stores the previous value of IEROPT in this variable. This can be used later to reset IEROPT.

FREQ uses ORACES internally to temporarily disable error termination and then reset IEROPT to the user's value. The user who wishes all parameters to have default values except IEROPT could use FREQNT and ORACES to accomplish this instead of using FREQNV.

Output

Output generated by FREQ is formatted into pages and written on the file with unit number NOUT which the user has set when initializing FREQ. Before initializing FREQ, the user must ensure that this file is available to the program by using the OPEN statement in FORTRAN or perhaps by using some host-machine-dependent file-handling protocol.

The output is formatted using column 1 for carriage control information. A "1" in column 1 means "print this line at the top of a new page," and a blank in column 1 means "advance the paper a single space from the last line and print this line;" no other carriage control characters are used. At the top of each page a heading is printed that includes the information stored in the parameter TITLE. The maximum number of lines written to a page is in parameter NLP (exception: if a matrix has so many columns that one row will not fit on a page, each row is written to the output file without a page break). Depending on the value of parameter NCL, lines will be formatted into a 132- or 80-character length. There is a small inconsistency in this use of the numbers "132" and "80". A 132-character line contains a carriage control character plus up to 132 additional characters. This is the capacity of a typical long-carriage line printer. An 80-character line contains a maximum of 79 characters including carriage control in an attempt to avoid triggering the extra carriage return (resulting in a blank line) which occurs when full 80-character lines are viewed on some 80-character-wide cathode-ray-tube (CRT) screens or printed on some 80-character printers.

User output may be interspersed with FREQ output on unit NOUT without confusing the page formatter *<u>if</u>* the user will inform FREQ of how many lines are to be written. This is done by issuing the FORTRAN statement

```
   CALL LNCNT(N)
```

just *<u>before</u>* issuing command(s) which cause the writing of N lines to unit NOUT. FREQ will move to a new page if necessary so that the block of N lines will be together. The user should both prefer making several calls with small values of N to making one call with a large value and avoid using $N > NLP - 2$ (two lines are used for the header). The user can also force subsequent writing to NOUT to start on a new page by using the FORTRAN statement:

```
   CALL LNCNT(-1)
```

FREQ will also move to a new page for writing done following a call to any of the initialization routines (except FREQNV with IOPT = −1).

Finally, all FREQ-generated output, except for error messages, can be turned on or off by user option when the FREQ computational interface routines described in the next section are called.

Error Processing

When FREQ detects an error, it writes a message to the file whose unit number is set in parameter NOUT and, if NERR $\neq$ NOUT, also to unit NERR. This allows the user to have a full output on one file and an error log on another (which might be an interactive terminal CRT screen). Whenever FREQ detects an error that is considered to be fatal (e.g., FREQ has failed to make some requested computation; almost all errors fall in this category) and if the parameter IEROPT is set to 0 (this is the default), FREQ stops (i.e., it stops using FORTRAN statement "STOP 'FREQ ERROR STOP' "). If the user has set IEROPT $\neq$ 0 or if the error is nonfatal, control returns to the user's program. Most FREQ computational subroutines set one or more error parameters that the user's program can test to see if FREQ detected any errors.

The user wishing to configure FREQ to minimize FREQ-generated output should OPEN a file with unit number NOUT, initialize FREQ with NERR = NOUT, and call FREQ computational interface routines with the output option parameter turned off (ISOPTA(3) = 0).

Use of Software: Computational Routines

Overview

There are five subroutines that constitute the user's interface to the computational routines of FREQ (as opposed to initialization, output, etc.). They are SIGCAL, SIGDYN, SIGTAB, SDTAB, and TABGEN.

SIGCAL performs a single-frequency analysis of the simpler system that results from removing the $\mathbf{K}_0$ block from figure 1(a). The "SIG" in the name comes from "sigma," the English spelling of the Greek letter commonly used to represent singular values of a matrix. "CAL" is for "calculation."

SIGDYN analyzes a system of the form of figure 1(a) at a single frequency. "SIG" is motivated as before, and "DYN" comes from the motivation for figure 1(a), which is a plant with dynamic compensator.

SIGTAB repeatedly calls SIGCAL to reanalyze the same system with a variety of frequency values. The principal outputs of SIGTAB are corresponding tabulations (hence the "TAB") of maximum and minimum singular values of the selected matrix as a function of frequency.

SDTAB functions analogously to SIGTAB except that it calls SIGDYN. The "SD" in SDTAB comes from SIGDYN; the "TAB" is motivated as before.

These are the driver routines for the computations that are FREQ's reasons for existence. In the following, references to "calling FREQ" should be understood to mean "calling one of the subroutines SIGDYN, SIGCAL, SIGTAB, or SDTAB."

TABGEN is a utility program that generates a list (table) of frequency values. The user supplies the first and last values, the number of points in the list, and a parameter that tells whether the values are to be equally spaced arithmetically (constant differences) or geometrically (constant ratios). The latter option is offered since graphs based on these tabulations commonly use a logarithmic scale for the frequency axis; and points in geometric progression become equally spaced on a log plot.

The user may either request SIGTAB or SDTAB to generate its own table of frequencies (which is done by a call to TABGEN) or the user may supply the table. One possibility would be for the user to call TABGEN to generate a spread of frequency values over a range of interest and then merge into that table additional points that refine the grid in the neighborhood of "interesting" frequencies (e.g., natural frequencies and transmission zeros). We illustrate this in the example presented in the section "Example" (see p. 21).

Figure 3 shows the relationships among the five interface routines with the user's software on the one hand and the remainder of the FREQ package on the other. Dotted lines indicate that the knowledgeable user can directly access those portions of the software, which have been taken from public sources. Instructions are available elsewhere (Laub 1981 and 1986; Dongarra et al. 1979; Smith et al. 1976) and will not be repeated here.

We now give the calling sequences and describe the arguments that are passed to the interface routines. TABGEN is a special case and will be described in full first. There is much overlap among the arguments passed to the other four routines, and so after giving the specific calling sequences we will present a unified discussion of these parameters.

TABGEN

Subroutine <u>**TABGEN**</u> generates a table of numbers spaced with the user's choice of equal increments or equal consecutive ratios. It is called by a statement of the form

```
CALL TABGEN(NO,OMGTAB,JOP,KERR)
```

The arguments in the calling sequence have the following meanings:

NO	an integer giving the number of elements in OMGTAB
OMGTAB	real array of at least NO locations. On entry, OMGTAB(1) and OMGTAB(NO) must be set to the first and last entries of the desired table. (If JOP = 2, these values must have the same sign.) On normal return, the remainder of the OMGTAB array is filled according to the option requested by parameter JOP.
JOP	an integer variable that the user sets to 0, 1, or 2 depending on the purpose of TABGEN. (On normal return, JOP = 0.) JOP = 0: virtually a "do nothing" option which is provided so that the call to TABGEN can be made in a loop; and if JOP is initialized outside the loop, only the first call will do the actual work of table generation. TABGEN does check that NO is positive, but it does not alter OMGTAB. JOP = 1: TABGEN fills in OMGTAB so that consecutive differences are equal. JOP = 2: TABGEN fills in OMGTAB so that consecutive ratios are equal.
KERR	an integer variable used to signal errors encountered by TABGEN. On normal return, KERR = 0. If an error is detected, KERR = −1 and OMGTAB and JOP are unchanged. Possible causes: NO non-positive; JOP not equal to 0, 1, or 2; or JOP = 2 but OMGTAB(1) and OMGTAB(NO) do not have the same sign.

Packed Arrays

All real and complex arrays passed between the user and FREQ need to be in packed format. By this we mean that the first column is stored in sequential array elements that are followed immediately (with no elements skipped) by the second column and so forth for each succeeding column. The software package ORACLS (Armstrong 1978a, 1978b, 1980) stores matrices in this fashion.

The principal danger to be avoided is passing a matrix that has been stored in an over-row-dimensioned array as in the following example:

```
   DIMENSION A(20,20)
   . . .
   DO 20 J = 1, 4
   DO 20 K = 1, 3
   <calculation resulting in the (K, J) matrix value>
   A(K,J) = <the calculated value>
20 CONTINUE
```

The 3×4 matrix stored in array A is not stored in packed format. For example, column 1 is stored sequentially, but then 17 array elements are skipped before the second column begins.

One way to correct this is to replace the DIMENSION statement by

```
DIMENSION A(3,n)
```

where $n \geq 4$. The critical thing is to have the first declared dimension equal the actual row count. Another correct technique is

```
  DIMENSION A(n)
```

where $n \geq 12$; then to set A, execute the code

```
      I = 0
      DO 20 J = 1, 4
      DO 20 K = 1, 3
      I = I + 1
      <calculation resulting in the (K, J) matrix value>
      A(I) = <the calculated value>
  20 CONTINUE
```

It is critical that the inner loop correspond to the first index. Otherwise, the transpose of the matrix will be stored in A.

Calling Sequences

With the parameters as defined below, these routines are called as follows:

```
  CALL SIGCAL(NG,LG,MG,AG,BG,CG,DG,DGNUL,IGOPTA,FIRSTG,SAVEG,
 1             EVRG,EVIG,IGERRA,RCONDG,
 2             OMEGA,RCONDS,ISOPTA,G,SIGMA,U,V,INFO,
 3             RDUM,CDUM,IDUM,JERR)
  CALL SIGDYN(NG,LG,MG,AG,BG,CG,DG,DGNUL,IGOPTA,FIRSTG,SAVEG,
 1             EVRG,EVIG,GO,IGERRA,RCONDG,
 2             NK,LK,MK,AK,BK,CK,DK,DKNUL,IKOPTA,FIRSTK,SAVEK,
 3             EVRK,EVIK,KO,IKERRA,RCONDK,
 4             OMEGA,RCONDS,ISOPTA,G,SIGMA,U,V,INFO,
 5             RDUM,CDUM,IDUM,JERR)
  CALL SIGTAB(NG,LG,MG,AG,BG,CG,DG,DGNUL,IGOPTA,FIRSTG,SAVEG,
 1             EVRG,EVIG,IGERRA,RCONDG,
 2             OMEGA,RCONDS,ISOPTA,G,SIGMA,INFO,
 3             RDUM,CDUM,IDUM,JERR,
 4             JOP,NO,OMGTAB,SVMAX,SVMIN,KERR)
  CALL SDTAB(NG,LG,MG,AG,BG,CG,DG,DGNUL,IGOPTA,FIRSTG,SAVEG,
 1             EVRG,EVIG,GO,IGERRA,RCONDG,
 2             NK,LK,MK,AK,BK,CK,DK,DKNUL,IKOPTA,FIRSTK,SAVEK,
 3             EVRK,EVIK,KO,IKERRA,RCONDK,
 4             OMEGA,RCONDS,ISOPTA,G,SIGMA,INFO,
 5             RDUM,CDUM,IDUM,JERR,
 6             JOP,NO,OMGTAB,SVMAX,SVMIN,KERR)
```

Under some of the options available, some of the parameters are not used. In this case, a dummy parameter may be used in the calling sequence. It is recommended that the standards of the American National Standards Institute (ANSI) be maintained in that such dummy parameters agree in type with what is expected. Thus, if in SIGCAL or SIGDYN the user elects ISOPTA(2) = 2, the U and V arrays are not used, but the position in the calling sequence must be occupied, and this position filler should be a complex array. CDUM could be used in both positions; or, for more safety, an array, say CD, could be declared as "COMPLEX CD(1)" and used.

In addition to the following discussion, program prelude documentation of these subroutines is given in appendix B. Here each subroutine is treated separately, and the parameters are explained in the order used in the subroutine call. This might be a useful tool for the programmer who is writing a code that uses FREQ.

Common Parameters: General

For the purposes of this discussion, we revert to the notation introduced in the section "Functional Description of Software" (see p. 3). The parameters naturally fall into three categories: (1) those involved with the calculation of $\mathbf{G}_0(j\omega)$ (and a parallel set for $\mathbf{K}_0(j\omega)$ if it is used), (2) those involved with the calculation of $\mathbf{G}_L(j\omega)$ and its singular values, and (3) (for SIGTAB and SDTAB only) those involved with controlling the repeated calculation for the several values of frequency. The parameters in each category can be further classified as those in which data are passed to FREQ or results are returned from FREQ (some parameters do both), parameters which specify options, and parameters which report error conditions.

The user is reminded that FREQ default handling of a fatal error is to write an error message and stop. The user who wishes to manage error processing differently must initialize FREQ with parameter IEROPT set to nonzero. Thereafter, if an error occurs, the error message will still be written, but control will return to the user's program which may then interrogate the error parameters.

Although there are a multitude of parameters provided for identifying possible error sightings by FREQ, the question of whether an error has occurred can be answered in a single test. If JERR = 0 (after SIGCAL or SIGDYN) or KERR = 0 (after SIGTAB or SDTAB), then no fatal errors were detected. Otherwise, further analysis of the error parameters is indicated.

Plant/Compensator Parameters

The option-setting parameters for the $\mathbf{G}_0(j\omega)$ calculation are IGOPTA and FIRSTG. IGOPTA is an integer array of at least two elements for SIGTAB and SDTAB or three elements for SIGCAL and SIGDYN. The third element is used to indicate if the user is calculating $\mathbf{G}_0(j\omega)$ externally; this option is not available to SIGTAB or SDTAB. The user sets IGOPTA(3) = 1 to indicate the external calculation of $\mathbf{G}_0(j\omega)$. Note that if IGOPTA(3) = 1, most of the plant parameters are ignored, including the first two entries in IGOPTA. The value of $\mathbf{G}_0(j\omega)$ is communicated to FREQ through parameter G in SIGCAL and G0 in SIGDYN. In this case, of the remaining $\mathbf{G}_0$ parameters, only LG and MG are used. The LG and MG are integer variables containing the number of rows and columns in $\mathbf{G}_0(j\omega)$. The G and G0 are complex matrices in which the user-calculated $\mathbf{G}_0(j\omega)$ is stored in packed format.

The remainder of the discussion of $\mathbf{G}_0(j\omega)$ parameters assumes that $\mathbf{G}_0(j\omega)$ is calculated internally by FREQ based on a linear, time-invariant state-space realization. This is always assumed by SIGTAB and SDTAB and is indicated to SIGCAL and SIGDYN by IGOPTA(3) = 0. (Any value for IGOPTA(3) except 0 or 1 generates a fatal error condition.) Then, $\mathbf{G}_0(j\omega)$ has the form

$$\mathbf{C}(j\omega\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}$$

Nonsingular matrices $\mathbf{T}$ exist such that $\mathbf{A}' = \mathbf{T}^{-1}\mathbf{A}\mathbf{T}$ is upper Hessenberg, i.e., zero below the first subdiagonal. (See, e.g., Golub and Van Loan (1983), p. 222, where $\mathbf{T}$ is chosen to be orthogonal.) Then, with

$$\mathbf{C}' = \mathbf{C}\mathbf{T} \qquad \mathbf{B}' = \mathbf{T}^{-1}\mathbf{B}$$

the result is

$$\mathbf{C}(j\omega\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D} = \mathbf{C}'(j\omega\mathbf{I} - \mathbf{A}')^{-1}\mathbf{B}' + \mathbf{D}$$

Performing the calculation in the latter form is much faster than the former since the matrix to be inverted is also upper Hessenberg (Laub 1981).

FIRSTG is a logical variable. The user sets FIRSTG = .TRUE. to indicate that the upper Hessenberg decomposition has not yet been done. FREQ will then do the decomposition, store the $\mathbf{A}'$, $\mathbf{B}'$, and $\mathbf{C}'$ matrices in the SAVEG array, and reset FIRSTG to ".FALSE.". Then, if FREQ is called with FIRSTG = .FALSE., FREQ does not repeat the upper Hessenberg decomposition but assumes that the reduced matrices are available in array SAVEG. Thus, once any FREQ routine has been called with given $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ matrices in a linear state-space realization with FIRSTG = .TRUE., the user can make future calls to the same or to a different FREQ routine with any $\mathbf{D}$ matrix, any frequency value(s), and any of the five choices of $\mathbf{G}(j\omega)$ matrices using the FIRSTG (which is now .FALSE.) and the SAVEG array returned when FREQ was called with FIRSTG = .TRUE..

THERE IS ONE EXCEPTION TO THIS USE OF FIRSTG: If SIGCAL or SIGTAB is called with DGNUL = .TRUE., signifying that the $\mathbf{D}$ matrix is all zeros, and ISOPTA(1) = 2, signifying that

$$\mathbf{G}(j\omega) = \mathbf{G}_0(j\omega)(\mathbf{I} + \mathbf{G}_0(j\omega))^{-1}$$

then an alternate formula for $\mathbf{G}(j\omega)$ is

$$\mathbf{G}(j\omega) = \mathbf{C}(j\omega\mathbf{I} - \mathbf{A} + \mathbf{BC})^{-1}\mathbf{B}$$

In this case FREQ bypasses the normal $\mathbf{G}_0(j\omega)$ calculation and calculates $\mathbf{G}(j\omega)$ directly by submitting the matrix $\mathbf{A}-\mathbf{BC}$ instead of $\mathbf{A}$ to the module that normally calculates $\mathbf{C}(j\omega\mathbf{I}-\mathbf{A})^{-1}\mathbf{B}$. This means that instead of reducing $\mathbf{A}$ to upper Hessenberg form, it is $\mathbf{A}-\mathbf{BC}$ that must be reduced; so a fresh decomposition (FIRSTG = .TRUE.) must be performed, and the user must realize that the resulting contents of SAVEG give an equivalent upper Hessenberg realization for the triple $(\mathbf{A}-\mathbf{BC}, \mathbf{B}, \mathbf{C})$ instead of for $(\mathbf{A}, \mathbf{B}, \mathbf{C})$.

IGOPTA selects options in the calculation of $\mathbf{G}_0(j\omega)$.

IGOPTA(1): This has an effect only when FIRSTG = .TRUE.. In that case, the possible values and their effects are as follows:

≥ 0: The $\mathbf{A}$ matrix (or $\mathbf{A} - \mathbf{BC}$ in the exceptional case) will be balanced and its eigenvalues determined using EISPACK routines BALANC and HQR (Smith et al. 1976) and returned to the user.

< 0: Do not do balancing and eigenvalue calculation.

The recommended choice is to set IGOPTA(1) ≥ 0. We discuss this further in the next section.

IGOPTA(2): Now let $(\mathbf{A}', \mathbf{B}', \mathbf{C}')$ be the equivalent upper Hessenberg realization for the triple $(\mathbf{A}, \mathbf{B}, \mathbf{C})$ (or for $(\mathbf{A} - \mathbf{BC}, \mathbf{B}, \mathbf{C})$ in the exceptional case) and set $\mathbf{H} = j\omega\mathbf{I} - \mathbf{A}'$. The value of IGOPTA(2) controls the sophistication of singular matrix detection applied to $\mathbf{H}$ in calculating $\mathbf{C}'\mathbf{H}^{-1}\mathbf{B}'$. Possible values and their effects are as follows:

< 0 : The reciprocal of the condition number of $\mathbf{H}$ is estimated and returned to the user in RCONDG. Then $\mathbf{H}$ is declared singular to working precision if the FORTRAN expression "1.0 + RCONDG .EQ. 1.0" evaluates as ".TRUE." and this is declared to be a fatal FREQ error.

≥ 0 : The only singularity checking is to see if any diagonal elements become 0.0 when $\mathbf{H}$ is factored by Gaussian elimination with partial pivoting.

IGOPTA(2) < 0 is recommended.

IGOPTA(3): This is used only by SIGCAL and SIGDYN. Its use is discussed in the first paragraph of this subsection.

NG, LG, MG, AG, BG, CG, DG, and **DGNUL** describe the linear state-space realization of $\mathbf{G}_0(s)$. The **A, B,** and **C** matrices are passed to FREQ in packed format in real arrays AG, BG, and CG. The integer in NG is the number of system states. This is also the row and column dimensions of AG, the row dimension of BG, and the column dimension of CG. The integer in LG is the number of system inputs. This is the column dimension of BG. MG contains the number of system outputs that is the row dimension of CG. DGNUL is a logical variable. A value of .TRUE. indicates that the system **D** matrix is null (all zeros). If DGNUL is .FALSE., the **D** matrix (MG $\times$ LG) is passed to FREQ in packed format in real array DG.

SAVEG is a real array of at least NG $\times$ (MG + NG + LG) elements. FREQ uses SAVEG to save the upper Hessenberg realization. The SAVEG array that is passed to FREQ with

FIRSTG = .FALSE. must be one that has been initialized by a prior call with FIRSTG = .TRUE.. See also the exceptional situation discussed above in connection with FIRSTG.

<u>**EVRG**</u> and <u>**EVIG**</u> are real arrays. If FIRSTG = .TRUE. and IGOPTA(1) ≥ 0, the real and imaginary parts of the eigenvalues of $\mathbf{A}$ (or $\mathbf{A} - \mathbf{BC}$ in the exceptional case) are returned to the user in EVRG and EVIG, respectively; so these must have at least NG elements each.

<u>**G0**</u> is a complex array of at least LG $\times$ MG elements used by SIGDYN and SDTAB to hold the $\mathbf{G}_0(j\omega)$ matrix which may be either internally calculated or, in the case of SIGDYN, precalculated by the user and passed to FREQ in G0.

<u>**IGERRA**</u> is an integer array of at least two elements that is used to report back to the calling program any errors detected in the $\mathbf{G}_0(j\omega)$ calculation.

IGERRA(1): This signals any errors in the eigenvalue calculation, so it is affected only if FREQ is called with FIRSTG = .TRUE. and ISOPTA(1) ≥ 0. The user is advised to set IGERRA(1) to 0 before first calling FREQ. Then, a return value of 0 signals either that the eigenvalue calculation detected no errors or that no eigenvalue calculation was called for. If the eigenvalue calculation is done, possible return values of IGERRA(1) are as follows:

= 0: Normal return.

> 0: Not all the eigenvalues have been computed. In this case, the values in EVRG and EVIG in index range 1, ..., IGERRA(1) are not to be trusted as eigenvalues of $\mathbf{A}$ (or $\mathbf{A} - \mathbf{BC}$); the remaining NG − IGERRA(1) were computed without errors being detected.

IGERRA(1) > 0 is considered a nonfatal error since FREQ makes no further use of these values.

IGERRA(2): This signals any error in calculating $\mathbf{C}'(j\omega\mathbf{I} - \mathbf{A}')^{-1}\mathbf{B}'$. Possible return values of IGERRA(2) are as follows:

= 0: Normal return.

= −1: FREQ was entered with IGOPTA(2) negative, indicating that the reciprocal condition number RCONDG of $j\omega\mathbf{I} - \mathbf{A}'$ was to be estimated, and the FORTRAN expression "1.0 + RCONDG .EQ. 1.0" evaluates as " .TRUE." so that the matrix is singular to working precision.

> 0 : FREQ was entered with IGOPTA(2) nonnegative, indicating that the calculation was to be done by using Gaussian elimination (with partial pivoting and without condition estimation) to factor $j\omega\mathbf{I} - \mathbf{A}'$ into the product of a permutation of a bidiagonal unit-diagonal lower triangular matrix and an upper triangular matrix. At location IGERRA(2), the diagonal of the upper triangular matrix is null, and thus $j\omega\mathbf{I} - \mathbf{A}'$ is singular.

A nonzero return is considered fatal. The user wishing to interrogate this parameter must initialize FREQ with IEROPT $\neq 0$; otherwise FREQ will generate an error stop before control is returned to the user's program.

<u>**RCONDG**</u> is a scalar variable in which FREQ stores the estimated reciprocal condition number of the $j\omega\mathbf{I} - \mathbf{A}'$ matrix used in calculating $\mathbf{G}_0(j\omega)$ when the user has elected this option by setting IGOPTA(2) < 0.

<u>**OMEGA**</u> is a real variable in which the frequency ω is stored at which $\mathbf{G}_0(j\omega)$ (and $\mathbf{K}_0(j\omega)$, if used) is evaluated when calling SIGCAL or SIGDYN. This variable is also passed to SIGTAB and

SDTAB where it is set internally from values in the array OMGTAB. On return, OMEGA contains the last frequency value at which calculation was attempted; this has potential diagnostic value in the event of an error exit from FREQ.

The parameters specifying $\mathbf{K}_0(j\omega)$ in SIGDYN and SDTAB parallel those for $\mathbf{G}_0(j\omega)$. For a description of these parameters, the reader should reread the "$\mathbf{G}_0(j\omega)$ calculation" paragraphs just preceding in which "$\mathbf{G}_0$" is replaced by "$\mathbf{K}_0$" and any "G" in an argument name is replaced by "K" (so that NG becomes NK, IGOPTA becomes IKOPTA, etc.)

The only limitations here are to ignore the OMEGA discussion (there is no "OMEKA" parameter) and to ignore anything referring specifically to SIGCAL and SIGTAB. In particular, eliminate any reference to storing a user-calculated transfer matrix in $\mathbf{G}$ and eliminate any reference to the exceptional $(\mathbf{A}-\mathbf{BC})$ calculation.

Loop-Gain Matrix (and Analysis Matrix) Parameters

The next group of parameters is concerned with the calculations that are made after $\mathbf{G}_0(j\omega)$ (and $\mathbf{K}_0(j\omega)$, if used) has become available to FREQ. For SIGDYN and SDTAB, this involves first determining the loop-gain matrix $\mathbf{G}_L(j\omega)$. (SIGCAL and SIGTAB use $\mathbf{G}_L = \mathbf{G}_0$.) Afterwards, the matrix $\mathbf{G}(j\omega)$ to be subjected to singular value analysis is calculated from $\mathbf{G}_L(j\omega)$, and finally the singular value analysis is performed. These parameters consist of several arrays used to store calculated values for internal use and/or to return them to the user (and, in one special case, input information into FREQ), an integer array for signaling options to FREQ, and several parameters for error reporting. The principal output parameters for SIGCAL and SIGDYN are included here. These same parameters are included in the calling sequences for SIGTAB and SDTAB, but they are reused for each frequency at which the calculation is being repeated so that the user sees only the values computed for the last value of frequency used. (The principal output parameters of SIGTAB and SDTAB are discussed below.)

ISOPTA is the option-determining parameter. It is an integer array of at least three elements for SIGCAL and SIGTAB or four elements for SIGDYN and SDTAB. ISOPTA(1) determines which matrix will be returned to the user in the G array. This is also the matrix subject to singular value decomposition, if elected.

ISOPTA(1): This has the additional function of determining whether the value in OMEGA or the values in OMGTAB are to be used as frequencies for continuous-system analysis or as normalized frequencies for discrete system analysis. The continuous case is discussed here; the discrete case is covered in the next section:

- = 1: $\mathbf{G} = \mathbf{G}_L$, the loop-gain transfer matrix.
- = 2: $\mathbf{G} = \mathbf{G}_L(\mathbf{I}+\mathbf{G}_L)^{-1}$, the feedback transfer matrix.
- = 3: $\mathbf{G} = (\mathbf{I}+\mathbf{G}_L)^{-1}$, the sensitivity matrix.
- = 4: $\mathbf{G} = \mathbf{I}+\mathbf{G}_L^{-1}$, the inverse return difference matrix.
- = 5: $\mathbf{G} = \mathbf{I}+\mathbf{G}_L$, the return difference matrix.

For all options except ISOPTA(1) = 1, $\mathbf{G}_L$ must be square.

ISOPTA(2): This determines which of three optional actions SIGCAL and SIGDYN will take with the $\mathbf{G}$ matrix:

- = 1: SIGCAL and SIGDYN return control to the user with no further calculation.
- = 2: The singular values of $\mathbf{G}$ are calculated and stored in the SIGMA array.
- = 3: In addition to calculating singular values as in option 2, the left and right singular vectors of $\mathbf{G}$ are calculated and stored in the U and V arrays.

Since the purpose of SIGTAB and SDTAB is to tabulate singular values of the $\mathbf{G}(j\omega)$ matrix as a function of ω, these routines overwrite the user's setting of ISOPTA(2) with option 2. (Option 1 would be pointless, and option 3 would involve wasted work.)

ISOPTA(3): This controls FREQ-generated output:

$\neq 0$: A considerable amount of output is written to logical unit NOUT (see the previous section on initialization, etc.) indicating the input data and the options being exercised, and also giving the results of many of the computations.

$= 0$: Output from FREQ is limited to error messages (if any). SIGTAB and SDTAB reset ISOPTA(3) to 0 after the first entry in the table of frequencies has been processed to limit what could otherwise be an overwhelming amount of output, much of it redundant.

ISOPTA(4): This determines how $\mathbf{G}_L$ will be formed from $\mathbf{G}_0$ and $\mathbf{K}_0$, so it is unnecessary for SIGCAL and SIGTAB since $\mathbf{G}_L$ is always $\mathbf{G}_0$ in these. In SIGDYN and SDTAB, the user selects which among $\mathbf{G}_0^{\pm 1}\mathbf{K}_0^{\pm 1}$ and $\mathbf{K}_0^{\pm 1}\mathbf{G}_0^{\pm 1}$ will be used for $\mathbf{G}_L$ by the following scheme:

Let $i_1 = 0$ or 1 depending on whether $\mathbf{G}_0$ or $\mathbf{G}_0^{-1}$, respectively, is to be used.

Let $i_2 = 0$ or 1 depending on whether $\mathbf{K}_0$ or $\mathbf{K}_0^{-1}$, respectively, is to be used.

Let $i_0 = 0$ or 1 depending on whether the order of multiplication is to be $\mathbf{G}_0^{\pm 1}\mathbf{K}_0^{\pm 1}$ or $\mathbf{K}_0^{\pm 1}\mathbf{G}_0^{\pm 1}$, respectively.

Finally, set ISOPTA(4) $= i_0 + 2i_1 + 4i_2$.

This implies that ISOPTA(4) is in the range of 0 through 7; other values create a fatal error in FREQ.

Let us denote the numbers of rows and columns of $\mathbf{G}_L$ by m and ℓ, respectively. Then, in SIGCAL and SIGTAB, m = MG and ℓ = LG. In SIGDYN and SDTAB, m and ℓ depend on i_0 . If $i_0 = 0$, then m = MG and ℓ = LK; whereas if $i_0 = 1, m$ = MK and ℓ = LG. Each choice of ISOPTA(4) imposes one or more obvious restrictions on the dimensions MG and LG of $\mathbf{G}_0$ and MK and LK of $\mathbf{K}_0$. (Dimensions must be compatible for forming the products; a matrix must be square to be inverted.)

<u>**G**</u> is a complex array of at least $m \times \ell$ elements (m and ℓ are defined in the previous paragraph). On normal return from FREQ, this contains, in packed format, the matrix $\mathbf{G}(j\omega)$ for ω = OMEGA (on return from SIGCAL or SIGDYN) or ω = OMGTAB(NO) (on return from SIGTAB or SDTAB). The G array is used as an input array in one circumstance; if SIGCAL is called with IGOPTA(3) = 1, the user-furnished value of $\mathbf{G}_0(j\omega)$ is passed to SIGCAL in array G in packed format.

<u>**SIGMA**</u> is a real array of at least min(m, ℓ) elements in which, if ISOPTA(2) ≥ 2, the nontrivial singular values of $\mathbf{G}(j\omega)$ (for the last ω value used, if SIGTAB or SDTAB were called) are stored.

<u>**U**</u> and <u>**V**</u> are complex arrays of at least $m \times$ min(m, ℓ) and $\ell \times$ min(m, ℓ) elements, respectively. If ISOPTA(2) = 3, the left and right singular vectors corresponding to the singular values in SIGMA are stored in the U and V arrays, respectively, in packed format. Thus, the columns of U and of V each form an orthonormal set of vectors and $\mathbf{G}(j\omega)$ = U $\times$ diag(SIGMA) $\times$ V(transposed).

<u>**RCONDS**</u> : Some of the options for forming $\mathbf{G}_L(j\omega)$ and $\mathbf{G}(j\omega)$ involve matrix inversion. In actual practice, each matrix inversion is coupled with a matrix multiplication either naturally or by introduction of an identity matrix and is reformulated as a solution of linear equations, with the matrix to be inverted becoming the coefficient matrix. This is accomplished using software from

LINPACK (Dongarra et al. 1979) which also estimates the reciprocal of the condition number of the coefficient matrix. The reciprocal condition number (numerically between 0.0 and 1.0) is returned to the user (for the last ω value used, if SIGTAB or SDTAB were called) in RCONDS. If more than one inversion is required, the minimal reciprocal condition number found is returned in RCONDS. If no inversion is called for, RCONDS is set to 2.0. If RCONDS is so small that the FORTRAN expression "1.0 + RCONDS .EQ. 1.0" evaluates as ".TRUE.", the coefficient matrix is singular to within machine precision and FREQ declares a fatal error.

INFO is an integer variable used to report singular value decomposition errors. If ISOPTA(2) = 2 or 3 (singular value decomposition is called for), INFO is set to the "INFO" parameter returned by the LINPACK (Dongarra et al. 1979) singular value decomposition routine; otherwise it is unchanged. The normal return value is 0. If INFO > 0, then the first min(m, ℓ) – INFO entries of SIGMA are singular values and the rest are undefined.

JERR is the "error summary" parameter for SIGCAL and SIGDYN and also passes error information for SIGTAB and SDTAB. On normal return, JERR = 0. Of the possible nonzero values, 2, 6, and 26 signal nonfatal errors; the rest are fatal. If both nonfatal and fatal errors are detected, the fatal error is reported in JERR (although the nonfatal error(s) still show up in the other error parameters). The possible values of JERR are as follows:

1. Matrix dimensions are nonpositive or inconsistent with calculations chosen by ISOPTA(1) or ISOPTA(4); or, ISOPTA(4) is out of range.
2. An error was detected in calculating the eigenvalues of the **A** (or $\mathbf{A} - \mathbf{BC}$) matrix in the $\mathbf{G}_0$ calculation. (See IGERRA(1) above.)
3. The matrix to be inverted in the $\mathbf{G}_0$ calculation is exactly singular; zero was detected on the diagonal of the upper triangular factor. (This corresponds to IGERRA(2) > 0.)
4. The matrix to be inverted in the $\mathbf{G}_0$ calculation is singular to working precision. (See the discussion of RCONDG and the case that IGERRA(2) = −1.)
5. IGOPTA(3) was neither 0 nor 1.
6. An error was detected in calculating the eigenvalues of the **A** matrix in the $\mathbf{K}_0$ calculation. (See IKERRA(1) above.)
7. An error was detected in calculating $\mathbf{K}_0$ that corresponds to the error signaled by JERR = 3 for $\mathbf{G}_0$.
8. An error was detected in calculating $\mathbf{K}_0$ that corresponds to the error signaled by JERR = 4 for $\mathbf{G}_0$.
9. IKOPTA(3) was neither 0 nor 1.
10. The matrix to be inverted in calculating $\mathbf{G}_L$ from $\mathbf{G}_0$ and $\mathbf{K}_0$ was singular to working precision. (See RCONDS above.)
11. The matrix to be inverted in calculating **G** from $\mathbf{G}_L$ was singular to working precision. (See RCONDS above.)
12. An error occurred in singular value decomposition. (See INFO above.)
26. Both nonfatal errors 2 and 6 above occurred.

IDUM, RDUM, and CDUM are three workspace arrays that must be provided: IDUM is of type integer, RDUM is of type real, and CDUM is of type complex. The minimum sizes of these arrays depend on a combination of factors involving the problem-sizing parameters NG, LG, and MG (and NK, LK, and MK if $\mathbf{K}_0$ is used) and which options are selected. For purposes of the following calculations, set NG (or NK) to 0 if the $\mathbf{G}_0(j\omega)$ (or $\mathbf{K}_0(j\omega)$) matrix is being user calculated. Recall that m and ℓ are the dimensions of the $\mathbf{G}_L(j\omega)$ matrix as defined following the discussion of ISOPTA(4) above.

IDUM must have at least max(m, NG) (SIGCAL, SIGTAB) or max(m, NG, NK) (SIGDYN, SDTAB) elements. RDUM must have at least NG (SIGCAL, SIGTAB) or max(NG, NK) (SIGDYN, SDTAB) elements.

The calculation for the minimum size of CDUM is quite complicated. However, a simple calculation gives an overestimate. Let $\underline{x}$ be the largest of NG, LG, and MG. Then CDUM needs at least

$$\underline{x}(\underline{x} + \text{MG} + \text{LG})$$

elements. If a compensator is used, repeat the calculation using K sizing parameters. If a larger number results, increase the size of CDUM to this new value. If this gives CDUM only three elements, increase to four. In the event that this is a SIGDYN or SDTAB application with ISOPTA(4) = 3 or 4, do the calculation for i_6 in the following paragraph and increase the size of CDUM, if necessary, to accommodate at least i_6 elements. In typical applications, this is not a very wasteful overestimate. For example, if the size of CDUM is determined by the first calculation using the plant-sizing parameters and if the plant has at least as many states (NG) as it has inputs (LG) or outputs (MG), then the number of unused elements in CDUM is less than the number of elements in the smaller of BG and CG.

The exact calculation for CDUM is given next. For use in SIGCAL and SIGTAB, the following calculation applies: Let i_1 = NG × (NG + LG + 1). If ISOPTA(1) = 2 and DGNUL = .FALSE., or if ISOPTA(1) = 3 or 4, let i_2 = MG × (2 × MG + 1); otherwise let $i_2 = 0$. If ISOPTA(2) = 2 or 3 and MG < LG, let i_3 = (MG + 1) × (LG + 2) − 1; otherwise let $i_3 = 0$. If ISOPTA(2) = 2 or 3 and MG ≥ LG, let i_4 = LG × (MG+2)+MG; otherwise let $i_4 = 0$. Then CDUM needs $\max(i_1, i_2, i_3, i_4)$ elements. For use in SIGDYN or SDTAB, the size of CDUM is calculated as follows: Let i_1 = NG × (NG + LG + 1) and i_2 = NK × (NK + LK + 1). If ISOPTA(1) = 2, 3, or 4, let $i_3 = m(2m+1)$; otherwise $i_3 = 0$. If ISOPTA(2) = 2 or 3 and $m < \ell$, let $i_4 = (m+1)(\ell+2) - 1$; otherwise $i_4 = 0$. If ISOPTA(3) = 2 or 3 and $m \geq \ell$, let $i_5 = \ell(m+2) + m$; otherwise $i_5 = 0$. Determine i_6 based on ISOPTA(4) from the following table:

ISOPTA(4)	i_6
0 or 1	0
2 or 5	$m(m+1)$
3 or 4	$\ell(2\ell+m+1)$
6 or 7	$m(2m+1)$

Then CDUM needs $\max(i_1, i_2, i_3, i_4, i_5, i_6)$ elements.

Tabulation Driver Parameters

Finally, there are the parameters used by SIGTAB and SDTAB to control the generation of tables of maximum and minimum singular values of the user's choice of **G** matrix. There is an option parameter, parameters defining the table of frequencies at which calculations will be performed, arrays to hold the resulting maximum and minimum singular values, and an error parameter.

<u>JOP</u> is an option parameter used to control internal generation of the table of frequencies OMGTAB at which calculations will be performed:

JOP = 0:	FREQ takes the current contents of the OMGTAB array as the table of frequency values.
JOP = 1:	FREQ fills in the interior NO-2 elements of OMGTAB based on what it finds in OMGTAB(1) and OMGTAB(NO) so that the entries are equally spaced.
JOP = 2:	FREQ fills in the interior NO-2 elements of OMGTAB based on what it finds in OMGTAB(1) and OMGTAB(NO) so that consecutive ratios of entries are equal. (These values will be equally spaced when plotted on a logarithmic scale.) For this option, OMGTAB(1) and OMGTAB(NO) must have the same sign.

FREQ returns JOP = 0 so that the same values can be used over without FREQ repeating the work of frequency-table generation.

<u>NO</u> (number of omegas) is an integer variable in which the user sets the number of entries in the arrays OMGTAB, SVMAX, and SVMIN.

<u>OMGTAB</u> (omega table) is a real array of at least NO elements which contains the frequencies at which calculations are to be performed. The user must either fully initialize OMGTAB and call

FREQ with JOP = 0 or initialize OMGTAB(1) and OMGTAB(NO) and call FREQ with JOP = 1 or 2.

<u>**SVMAX**</u> and <u>**SVMIN**</u> (singular value maximum, etc.) are real arrays of at least NO elements each. On normal exit, SVMAX(I) and SVMIN(I) contain the maximum and minimum singular values of the user-selected $\mathbf{G}(j\omega)$ matrix for ω = OMGTAB(I), I = 1, ..., NO.

<u>**KERR**</u> is the error-summary parameter for SIGTAB and SDTAB. The normal return is 0. If FREQ returns KERR = −1, it has detected an error in the input values of NO (e.g., negative), JOP (none of 0, 1, or 2), or OMGTAB (end points of different signs when JOP = 2). If FREQ returns KERR > 0, it means that a fatal error has been detected while attempting the requested calculation with OMEGA = OMGTAB(KERR). In this case, JERR, the more detailed error parameters, and/or the output or error file should be consulted to determine the exact location of the error. Even if KERR = 0, the user should consult JERR before trusting any returned eigenvalues (EVRG and EVIG; and, if used, EVRK and EVIK).

Use of Software: Discrete System Analysis

The singular-value-based loop-shaping methodology considered until now for transfer matrices arising from continuous, linear time-invariant systems also applies in the discrete system case. In the discrete case, transfer matrices are obtained through z-transform theory; and in the control design and system analysis process, the same family (loop gain, return difference, closed-loop response, inverse return difference, and sensitivity) of transfer matrices is important. The only structural difference in the discrete transfer matrices is the appearance of the z-transform parameter z in place of the Laplace transform parameter s.

Discrete frequency-response calculations, such as those considered by FREQ, replace z by $e^{j\bar{\omega}}$ where $\bar{\omega}$ is the normalized frequency, also called the normalized angular frequency (Kwakernaak and Sivan 1972). The choice of the set of values for $\bar{\omega}$ is based on particular discrete system characteristics such as sampling interval and Nyquist frequency (Oppenheim, Willsky, and Young 1983), and, as in the continuous case, the choice will be left to the user.

Although FREQ will accept any value for $\bar{\omega}$, all available information about the singular values of any of the transfer matrices in the FREQ family can be obtained by limiting values of $\bar{\omega}$ to $[0, \pi]$. In the first place, $e^{j\bar{\omega}}$ is a periodic function of $\bar{\omega}$ of period 2π, and so $\bar{\omega}$ may be limited to $[-\pi, \pi]$. In the second place, $e^{-j\bar{\omega}}$ is the complex conjugate of $e^{j\bar{\omega}}$, and so any discrete transfer matrix of the FREQ family calculated from $-\bar{\omega}$ is the complex conjugate of the matrix calculated from $\bar{\omega}$, and these have the same singular values. This means that information calculated from choosing $\bar{\omega}$ in $[-\pi, 0]$ (or, by periodicity, in $[\pi, 2\pi]$) duplicates information calculated from $\bar{\omega}$ in $[0, \pi]$.

In order to access the FREQ discrete system analysis capability, the user simply follows all the instructions for the continuous case with two exceptions:

1. The frequency values input to FREQ in variable OMEGA or in array OMGTAB should be reasonable as normalized frequencies.
2. The user must add "10" to the value given ISOPTA(1), the transfer matrix choice parameter.

For purposes of applying the instructions of the previous section to the discrete calculation, the user should ignore the "10" added to ISOPTA(1) in step 2 above. Stated another way, if one wants to analyze, say, the feedback transfer matrix of a discrete system, then one must set ISOPTA(1) = 12, but follow any special instructions given in the previous section for ISOPTA(1) = 2.

Software Notes

This section goes into more detail about the code and the algorithms.

It describes in which language the code is written, where it is available, and what parts of it have been obtained from other sources. It then discusses the calculations done by FREQ and tells what is being calculated, how it is being calculated, and what can go wrong. This will hopefully help a user diagnose the source of trouble in the event that FREQ detects an error.

Notes on the Code

FREQ exists in single- and double-precision versions. The single-precision version is written in ANSI standard FORTRAN 77. The double-precision version fails to be ANSI standard only in its use of data-type COMPLEX*16 and associated intrinsic functions (e.g., DCMPLX). It is believed that any dialect of FORTRAN 77 that can handle the complex double-precision LINPACK (that is, the LINPACK routines whose names start with "Z"; Dongarra et al. 1979) can also compile double-precision FREQ. FREQ is intended to be portable code. The single-precision version has been exercised on Control Data Corporation (CDC) CYBER computers under the NOS operating system, on Digital Equipment Corporation (DEC) VAX computers under the VMS system, and on Sun Microsystems computers under the UNIX operating system; and the double-precision version has been exercised on VAX and Sun. Based on these tests, the use of double precision on 32-bit machines is recommended.

The authors had the following problem with the version of LINPACK (Dongarra et al. 1979) used in this work. The associated Basic Linear Algebra Subroutines (BLAS) contain function subprograms whose FUNCTION statements have the form

```
COMPLEX FUNCTION ZXXXX*16(...)
```

Some compilers (e.g., f77 under UNIX on Sun minicomputers) do not recognize this syntax. If this is rewritten as

```
COMPLEX*16 FUNCTION ZXXXX(...)
```

it compiles both on the Sun and on VAX using the VMS FORTRAN compiler.

Source code and documentation (a copy of this TM) are available from COSMIC (NASA's software dissemination center):

Computer Software Management and Information Center (COSMIC)
The University of Georgia
382 East Broad Street
Athens, GA 30602

Ordering information may be found in the current COSMIC catalog; at the time that this is being written, the current catalog is COSMIC (1989). Both the single- and double-precision versions are available in brief and complete form. The brief forms fail to be complete in that the LINPACK and EISPACK modules have been omitted. These versions provide more economical copies of the code that can be used by users who have alternate access to the standard software packages LINPACK and EISPACK. In the further interest of economy, comment lines have been removed from all versions of the COSMIC source code. Some of these have been included in this document (appendix B).

COSMIC identifies the four versions of FREQ as follows:

LAR-14119	single precision; LINPACK and EISPACK modules omitted
LAR-14120	single precision; complete
LAR-14121	double precision; LINPACK and EISPACK modules omitted
LAR-14122	double precision; complete

In constructing the software package FREQ, code has been incorporated from other sources. In addition to the aforementioned LINPACK (Dongarra et al. 1979) and EISPACK (Smith et al. 1976) modules, FREQ uses the efficient, multivariable frequency-response software of Laub (1981, 1986) that was first published in Lehtomaki (1981). Laub's software has been published (Laub 1986) under copyright, 1986, by the Association for Computing Machinery (ACM), Inc., and has been incorporated into FREQ for distribution by COSMIC by permission of the ACM. The authors are grateful to the ACM for the permission and to Laub for furnishing his double-precision version.

As with any subroutine package, the user should avoid naming any of his global entities (program name, subroutine names, common block names, etc.) with symbolic names used by FREQ for global entities. A list of these is given in appendix C. If the subroutine package ORACLS (Armstrong 1978a, 1978b, 1980) or any of its derivatives is also being used, there is a possibility of conflict since some of the symbolic names used for global entities in FREQ are also used in ORACLS. The user wishing to combine FREQ and ORACLS or a derivative in a single application has the best chance of success if the following procedures are observed: Link (load) the software in such a manner that if a subroutine or common block name occurs both in FREQ and ORACLS, the version from

FREQ is used. Execute any initialization protocols for ORACLS first, and then perform the FREQ initialization. The user is warned that some developmental versions of ORACLS are incompatible with FREQ in that common blocks are used that have the same name but different lengths.

Notes on the Algorithm

The subroutines SIGTAB and SDTAB perform data and parameter value management and error detection. If the user requests generation of an array of frequency values, this is done by a call to TABGEN. The calculations of transfer function matrices, the loop-gain matrix, etc., and singular values are accomplished by repeatedly calling SIGCAL or SIGDYN, respectively, for each frequency in OMGTAB.

On entry to SIGCAL or SIGDYN, the first task performed is to check if the integer parameters that give the system size are in proper range and are consistent for the calculations selected. It is verified that all matrix dimensions are positive, that the sizes are compatible if any matrix multiplication is called for, and that any matrix to be inverted or added to an identity matrix is square. The error signal JERR = 1 signals the failure of one of these tests.

Once again, the discussion will focus on the continuous case with transfer matrices being evaluated at $j\omega$. These remarks also apply to the discrete case by replacing $j\omega$ with $e^{j\omega}$.

Unless the user signals that the $\mathbf{G}_0(j\omega)$ (or $\mathbf{K}_0(j\omega)$ in SIGDYN) matrix has been precalculated, the next job is to perform this calculation. Only the $\mathbf{G}_0(j\omega)$ calculation is discussed; the $\mathbf{K}_0(j\omega)$ calculation is analogous (in fact, it is performed by a call to the same subroutine). There are several alternatives in this calculation, and they are controlled principally by the values of FIRSTG, IGOPTA(1), IGOPTA(2), and DGNUL with some possible influence (in SIGCAL only) by ISOPTA(1).

If FIRSTG = .TRUE., FREQ calculates an equivalent realization for the linear system with realization $(\mathbf{A}, \mathbf{B}, \mathbf{C})$ (or $(\mathbf{A}-\mathbf{BC}, \mathbf{B}, \mathbf{C})$ in the exceptional case) where the matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are in arrays AG, BG, and CG, respectively. Recall that the term "exceptional case" refers to the case that SIGCAL has been called with DGNUL = .TRUE. and ISOPTA(1) = 2. The equivalent realization is characterized by having an upper Hessenberg (i.e., null below the first subdiagonal) system matrix. The inclusion of some optional steps in the Hessenberg decomposition is controlled by IGOPTA(1).

First, the matrices $\mathbf{A}$ (or $\mathbf{A} - \mathbf{BC}$ in the exceptional case), $\mathbf{B}$, and $\mathbf{C}$ are copied from the user's AG, BG, and CG arrays (which are never changed by FREQ) and packed into the SAVEG array. If IGOPTA(1) $\geq$ 0, the copy of the system matrix in the SAVEG array is balanced using EISPACK subroutine BALANC (Smith et al. 1976; this is a row-and-column equilibration procedure that also isolates eigenvalues by row-and-column permutations where possible), and the appropriate transformations are applied to the $\mathbf{B}$ and $\mathbf{C}$ matrices so as to maintain equivalence of representation. The eigenvalues of the system matrix are then computed using EISPACK subroutine HQR. FREQ returns these eigenvalues to the user but makes no further use of them. Laub recommends that the balancing and eigenvalue computation be done. In a private communication, he said "Eigenvalue computation ... is recommended mainly to get an idea of where the poles are (i.e., where the frequency response can be expected to shoot off to infinity). Balancing is done to enhance the accuracy of computed eigenvalues and has to do with the backward error analysis of the QR algorithm." One would expect that the balancing would also be numerically beneficial to the transformation into upper Hessenberg form.

Whether the system matrix has been balanced or not, the next computation is to use orthogonal similarity transformations to reduce the system matrix to upper Hessenberg form, accumulating the transformations into the $\mathbf{B}$ and $\mathbf{C}$ matrices so that the linear system represented by the matrices stored in array SAVEG remains equivalent to the original system. The variable FIRSTG is set to ".FALSE." so that the user can recall SIGCAL or SIGDYN with the same FIRSTG and SAVEG without doing anything else to them.

The $\mathbf{G}_0(j\omega)$ calculation continues with the calculation of $\mathbf{C}(j\omega\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}$. (This point is where the calculation starts if FREQ were entered with FIRSTG set .FALSE.; in the exceptional case, $\mathbf{C}(j\omega\mathbf{I} - \mathbf{A} + \mathbf{BC})^{-1}\mathbf{B}$ is calculated; further remarks are limited to the normal case.) Now the matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are taken from array SAVEG, and so $\mathbf{A}$ is upper Hessenberg and therefore $j\omega\mathbf{I} - \mathbf{A}$ is also. The term $(j\omega\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}$ is calculated by solving the matrix equation $(j\omega\mathbf{I} - \mathbf{A})\mathbf{Z} = \mathbf{B}$

for **Z**. Laub's software does this by using subroutines patterned after the LINPACK general equation solvers, but taking advantage of the upper Hessenberg structure of the coefficient matrix. As with LINPACK, the user has the choice of whether or not to estimate the reciprocal condition number of the coefficient matrix. The former, which is signaled to FREQ by setting IGOPTA(2) < 0, is the more robust means of detecting whether or not the coefficient matrix is so ill-conditioned that the results are unreliable, and it is the generally recommended method. If the open-loop eigenvalues are all sufficiently separated from the point(s) $j\omega$ at which $\mathbf{G}_0(j\omega)$ is to be calculated, the faster IGOPTA(2) ≥ 0 may be elected. In either case, an attempt is made to detect ill-conditioning or singularity in the coefficient matrix. Results of this attempt are reported back to the user in parameters RCONDG and IGERRA(2) as discussed in the previous section.

Next, if DGNUL = .FALSE., the calculation of $\mathbf{G}_0(j\omega)$ is completed by adding the matrix in the DG array to the matrix calculated above.

In SIGDYN, after $\mathbf{G}_0(j\omega)$ and $\mathbf{K}_0(j\omega)$ have been calculated, the next task is to form the loop-gain matrix $\mathbf{G}_L(j\omega)$. This step does not apply to SIGCAL since it takes $\mathbf{G}_L(j\omega) = \mathbf{G}_0(j\omega)$. Usually, forming the loop-gain matrix will involve only a matrix multiplication. However, some options do involve matrix inverses, which can give numerical problems. In this case, $\mathbf{G}_L$ is calculated by formulating the calculation as the solution of a linear system and using LINPACK (Dongarra et al. 1979) subroutines for solving systems of equations with general complex-valued coefficient matrices. For example, if

$$\mathbf{G}_L(j\omega) = \mathbf{G}_0(j\omega)\ \mathbf{K}_0(j\omega)^{-1}$$

the linear system $\mathbf{K}_0'\mathbf{G}_L' = \mathbf{G}_0$ is solved for $\mathbf{G}_L'$ (the transpose of $\mathbf{G}_L$), and thus $\mathbf{G}_L$ is determined. If

$$\mathbf{G}_L(j\omega) = \mathbf{G}_0(j\omega)^{-1}\ \mathbf{K}_0(j\omega)^{-1}$$

$(\mathbf{K}_0\mathbf{G}_0)\mathbf{G}_L = \mathbf{I}$ is solved for $\mathbf{G}_L$, where $\mathbf{I}$ is an identity matrix. If a linear system is solved, then the reciprocal condition number of the coefficient matrix is estimated, returned to the user in parameter RCONDS (unless a worse value is generated later), and a fatal error is declared if RCONDS is so small that the FORTRAN expression "1.0+RCONDS.EQ.1.0" evaluates as ".TRUE.".

It was decided not to include the analogous option in SIGCAL that would allow $\mathbf{G}_L(j\omega) = \mathbf{G}_0(j\omega)^{-1}$. Almost all the results that would be added by offering this option can be obtained by taking note of the following observations: The closed-loop response matrix for $\mathbf{G}_0^{-1}$ is the sensitivity matrix for $\mathbf{G}_0$, and conversely. The return difference matrix for $\mathbf{G}_0^{-1}$ is the inverse return difference matrix for $\mathbf{G}_0$, and conversely. The singular values of $\mathbf{G}_0^{-1}$ are the reciprocals of the singular values of $\mathbf{G}_0$.

FREQ next calculates the matrix selected by the user in ISOPTA(1). If ISOPTA(1) = 1, there is nothing to do and control passes to the next step. In SIGCAL, FREQ is mindful at this point of the exceptional case, and if ISOPTA(2) = 2 and DGNUL = .TRUE., SIGCAL assumes that the linear system transfer function calculation has been done with the linear system matrix replaced by $\mathbf{A} - \mathbf{BC}$ so that no further calculation is needed to obtain the feedback transfer matrix. Otherwise, identity matrices are added and/or linear systems are solved to obtain an inverse or the product of the inverse of a matrix with another matrix as required. The same module for calling LINPACK linear equation solvers is used here as was used in the $\mathbf{G}_L$ calculation in SIGDYN, and so the same remarks about error detection apply.

Finally, if the user has elected this option, singular values and, if chosen, singular vectors of $\mathbf{G}_L(j\omega)$ are calculated. This is done by the single-precision or double-precision subroutine for singular value decomposition of a complex matrix from LINPACK (Dongarra et al. 1979). If any errors are made at this point, they are signaled in parameter INFO which FREQ copies from LINPACK. The user encountering errors at this point and wishing to understand them should refer to the discussion of INFO by Dongarra et al. (1979, see p. 11.5). The arrays S and E referred to there are located, respectively, in the first min$(m + 1,\ \ell)$ and the next min$(m,\ \ell)$ locations of work array CDUM. See the discussion of ISOPTA(4) in the section "Use of Software: Computational Routines" (see p. 7) for the definitions of m and ℓ.

Example

Use of the FREQ code is demonstrated with numerical data taken from NASA TP-2560 (Joshi, Armstrong, and Sundararajan 1986) wherein a linear-quadratic-Gaussian (LQG)/loop-transfer-recovery (LTR) controller synthesis procedure is applied to the problem of designing a fine-pointing attitude control system for a large flexible antenna. The study is based on a 26th-order finite-element model of the 122-m hoop/column antenna, which consists of three rigid-body rotational modes and the first 10 elastic modes. A 12th-order compensator design model is formed by taking the rigid body and the first three elastic modes. The remaining truncated elastic mode dynamics are interpreted as an additive perturbation to the design system and are used as an approximate representation of the unmodeled dynamics for use in stability robustness tests. The LQG/LTR procedure produced estimator and controller gain matrices defining a 12th-order LQG-type compensator.

We use the 26th-order model of the hoop/column antenna as given in the appendix of NASA TP-2560 and refer to its transfer matrix as $\mathbf{G}$. Truncation of this system to its first 12 states gives the design model (plant) $\mathbf{G}_p$. Truncation of this same 26th-order model to its last 14 states gives the additive uncertainty model $\Delta\mathbf{G}$. We use the estimator (filter gain) and control gain matrices given in table II of TP-2560 together with the linear system representation of $\mathbf{G}_p$ in the last formula on page 5 of TP-2560 to construct a linear system model of the controller; its transfer matrix is denoted by $\mathbf{G}_c$.

The table of frequency values used by SIGTAB and SDTAB is generated by merging the table used in generating the plot data for TP-2560 (containing selected open-loop eigenvalue and invariant zero frequencies) with a grid of values giving 100 intervals per decade spaced in a geometric fashion. This illustrates the technique suggested in the earlier discussion of TABGEN. We then tabulate the maximum and minimum singular values of $\mathbf{G}$, $\mathbf{G}_p$, $\Delta\mathbf{G}$, and $\mathbf{G}_c$. Frequency domain analysis is then applied to the plant-compensator combinations $\mathbf{G}_p\mathbf{G}_c$ and $\mathbf{G}\mathbf{G}_c$. Plots of the resulting tabulations are presented in figures 4–14 with the maximum and minimum singular values labeled $\sigma_{\max}$ and $\sigma_{\min}$, respectively.

Figure 4 presents the open-loop response of the full 26th-order model of the hoop/column antenna ($\mathbf{G}$). Figure 5 shows the same information for the design model ($\mathbf{G}_p$). This corresponds to figure 18 in TP-2560. Figure 6 shows the frequency response for the additive error model ($\Delta\mathbf{G}$), corresponding to figure 19 in TP-2560. Figure 7 shows the same information for the compensator ($\mathbf{G}_c$). Figures 8 and 9 refer to the design model with compensator ($\mathbf{G}_p\mathbf{G}_c$). Figure 8 shows the open-loop response (corresponding to fig. 23 in TP-2560), and figure 9 shows the closed-loop response.

Finally, figures 10–14 apply the full spectrum of FREQ analysis to the transfer matrix $\mathbf{G}\mathbf{G}_c$, representing the compensator applied to the full 26th-order model of the hoop/column antenna. Figure 10 shows the loop-gain transfer matrix (open-loop response), figure 11 shows the feedback transfer matrix (closed-loop response), figure 12 shows the sensitivity matrix, figure 13 shows the inverse return difference matrix, and figure 14 shows the return difference matrix.

A listing of the code used to produce this example is given in appendix D. This was run on a CDC CYBER 180-series computer under the NOS operating system. Each single-precision floating-point number occupies 60 binary bits with 48 bits being used for the mantissa. The subroutines READ, TRANP, UNITY, NULL, and READ1 used by this example are identical in function (except in case an error is detected) with the subroutines of the same name in subroutine package ORACLS (Armstrong 1978a, 1978b, 1980). They have been modified to be compatible with FREQ input/output (I/O) and error processing.

The resulting output is given in appendix E. The input data are not listed separately since they are echoed as the first five matrices printed in the output listing.

NASA Langley Research Center
Hampton, VA 23665-5225
July 10, 1989

Appendix A

Guide to Singular Value Tabulation

This appendix gives an annotated listing of the subroutine SDTAB. This can serve as a template for the user who wishes to drive FREQ to do frequency analysis on a linear system at several values of frequency, but cannot fit the task within the restrictions placed on the use of SIGTAB and SDTAB.

```
      SUBROUTINE SDTAB(NG,LG,MG,AG,BG,CG,DG,DGNUL,IGOPTA,FIRSTG,SAVEG,
     1                 EVRG,EVIG,GO,IGERRA,RCONDG,
     2                 NK,LK,MK,AK,BK,CK,DK,DKNUL,IKOPTA,FIRSTK,SAVEK,
     3                 EVRK,EVIK,KO,IKERRA,RCONDK,
     4                 OMEGA,RCONDS,ISOPTA,G,SIGMA,INFO,
     5                 RDUM,CDUM,IDUM,JERR,
     6                 JOP,NO,OMGTAB,SVMAX,SVMIN,KERR)
      COMPLEX
     + CDUM(*), G(*), GO(MG,*), KO(MK,*)
      INTEGER
     + JERR, JOP, KERR, MG, MK, NG, NK, NO, LG, LK, IDUM(*), IGERRA(2),
     + IGOPTA(2), IKERRA(2), IKOPTA(2), INFO, ISOPTA(4)
      LOGICAL
     + DGNUL, DKNUL, FIRSTG, FIRSTK
      REAL
     + OMEGA, RCONDG, RCONDK, RCONDS, AG(*), AK(*), BG(*), BK(*), CG(*),
     + CK(*), DG(*), DK(*), EVIG(*), EVIK(*), EVRG(*), EVRK(*),
     + OMGTAB(*), RDUM(*), SAVEG(*), SAVEK(*), SIGMA(*), SVMAX(*),
     + SVMIN(*)
```

The purpose of SDTAB is to repeatedly call SIGDYN in its "calculate both G0 and K0 internally" mode while varying the value of the frequency parameter OMEGA so as to tabulate the maximum and minimum singular values of the **G** matrix which SIGDYN chooses to calculate based on the value of the parameter "ISOPTA(1)". These comments assume that the reader is familiar with the description of how to use SIGDYN. (See the section "Use of Software: Computational Routines" (see p. 7) and/or the program prelude documentation of SIGDYN found in appendix B.)

A running commentary is interspersed throughout the code to aid the user who wishes to write a routine to tabulate singular values outside the restrictions placed on SIGTAB and SDTAB, such as for a system with one or both of the G0 and K0 matrices calculated externally.

Variables for internal use are declared:

```
      CHARACTER*77 MSG
      COMPLEX
     + U(1), V(1)
      INTEGER
     + I, IGO(3), IKG, IKO(3), L, LEROPT, LERR, M, NEROPT, NSIG
```

The IGO and IKO arrays are used to force the "calculate G0 and K0 internally" option in SIGDYN without altering the user's IGOPTA or IKOPTA. The user writing a tabulation driver with external calculations of either G0 or K0 must make appropriate changes here:

```
      IGO(1) = IGOPTA(1)
      IGO(2) = IGOPTA(2)
      IGO(3) = 0
      IKO(1) = IKOPTA(1)
      IKO(2) = IKOPTA(2)
      IKO(3) = 0
      KERR = -1
```

Generate an error return if NO is not positive:

```
      IF (NO .LE. 0) RETURN
```

Bypass table generation if there are only one or two OMEGA's:

```
      IF (NO .LE. 2) GO TO 100
```

Bypass table generation if the OMGTAB array is already filled:

```
      IF (JOP .EQ. 0) GO TO 100
```

Save the user's value of error option parameter "IEROPT" in "LEROPT" and set IEROPT to 1 so that fatal errors will not cause a program stop. The user's wishes, as expressed in the setting of IEROPT, will ultimately be honored; but, by temporarily overriding that setting, if there is an error, the user will discover not only that it occurred in TABGEN but also that it was at a time when TABGEN was called from SDTAB. This overriding and then resetting of IEROPT is done again around the call to SIGDYN so that any errors detected there will be reported to the user as coming from a call to SIGDYN made by SDTAB:

```
      NEROPT = 1
      CALL ORACES(NEROPT,LEROPT)
```

Generate the table of frequencies:

```
      CALL TABGEN(NO,OMGTAB,JOP,KERR)
```

Now reset IEROPT to the user's value:

```
      CALL ORACES(LEROPT,NEROPT)
```

If no error, go on:

```
      IF (KERR .EQ. 0) GO TO 100
```

If there was an error, write a "FATAL ERROR" message:

```
      CALL ORCERP(3,'ERROR IN TABGEN CALLED FROM SDTAB')
```

Depending on the user's setting of IEROPT, this will either cause a "FREQ ERROR STOP" program stop or will return control at the following line after writing out the error message:

```
      RETURN
```

Normal processing continues at this point:

```
  100 CONTINUE
      JOP = 0
```

Turn off generation of the **U** and **V** matrices of the singular value decomposition (SVD):

```
      ISOPTA(2) = 2
```

Determine the end of the main diagonal (NSIG) of the SVD diagonal matrix (code through "30 CONTINUE" not needed for a SIGCAL driver):

```
      IKG = MOD(ISOPTA(4),2)
```

If IKG .EQ. 1, then M = MK and L = LG; otherwise, M = MG and L = LK:

```
      M=MK
      L=LG
      IF (IKG .NE. 0) GO TO 30
      M=MG
      L=LK
   30 CONTINUE
```

If the user is writing a SIGCAL driver, replace the following line by "NSIG = MIN(MG,LG)":

```
      NSIG = MIN(M,L)
```

Begin the loop to calculate singular values as a function of OMEGA. Initialize JERR:

```
      JERR = 0
      DO 20 I=1,NO
```

In case of error on this OMEGA, KERR will be set:

```
      KERR = I
```

Pick up the current value of OMEGA:

```
      OMEGA = OMGTAB(I)
```

<u>**THIS IS THE PLACE**</u> to do any external calculation of G0 or K0 matrices (G for SIGCAL). Note also the correspondence between the parameters in the following call and those of this subroutine. The user writing a SIGCAL (or SIGDYN) driver code would probably want to use IGOPTA (and IKOPTA) in place of IGO (and IKO) below.

Save the user's value of error option parameter "IEROPT" in "LEROPT" and set IEROPT to 1 so that fatal errors will not cause a program stop. (See the remarks surrounding the similar code near the call to TABGEN above.) Thus:

```
      NEROPT = 1
      CALL ORACES(NEROPT,LEROPT)
```

Determine the system matrix, etc., and singular values:

```
      CALL SIGDYN(NG,LG,MG,AG,BG,CG,DG,DGNUL,IGO,FIRSTG,SAVEG,
     1            EVRG,EVIG,GO,IGERRA,RCONDG,
     2            NK,LK,MK,AK,BK,CK,DK,DKNUL,IKO,FIRSTK,SAVEK,
     3            EVRK,EVIK,KO,IKERRA,RCONDK,
     4            OMEGA,RCONDS,ISOPTA,G,SIGMA,U,V,INFO,
     5            RDUM,CDUM,IDUM,LERR)
```

Now reset IEROPT to the user's value:

```
      CALL ORACES(LEROPT,NEROPT)
```

This suppresses printout after the first OMEGA. (Printout for OMGTAB(1) is determined by user initialization of ISOPTA(3).) Thus:

```
      ISOPTA(3)=0
```

If any error was detected, update JERR. (By this buffering of error codes, the nonfatal error code will be reported to the user if a nonfatal eigenvalue calculation error is detected on the first OMEGA and no other error is found. If a fatal error is found later, it will override the nonfatal error code in JERR.) Thus,

```
      IF (LERR .NE. 0) JERR = LERR
```

If a fatal error was detected, abandon the calculation:

```
      IF (JERR .NE. 0 .AND. JERR .NE. 2 .AND. JERR .NE. 6 .AND.
     + JERR .NE. 26) GO TO 40
```

Tabulate the maximum and minimum singular values. Any other "once per OMEGA" code could go here (e.g., brief printout, tabulation of other quantities, etc.). This is the end of the tabulation loop:

```
      SVMAX(I) = SIGMA(1)
      SVMIN(I) = SIGMA(NSIG)
   20 CONTINUE
```

If execution reaches this point, the singular values have been calculated for all OMEGA's in OMGTAB without any error being detected. Generate a "normal" return:

```
      KERR = 0
      RETURN
```

This is where control passes if any fatal error is detected in the loop that works through the values in OMGTAB. A "fatal error" message will be written and, depending on the user's setting of IEROPT, either a "FREQ ERROR STOP" will be generated or control will return to the calling program:

```
   40 CONTINUE
      WRITE (MSG,50) KERR,OMEGA
   50 FORMAT('ERROR IN SDTAB, KERR = ',I7,'  AT OMEGA = ',1PE13.6)
      CALL ORCERP (3,MSG)
      RETURN
      END
```

Appendix B

Selected Subroutine Preambles

This appendix contains the program preambles from the five user interface routines from FREQ (i.e., SIGCAL, SIGDYN, SIGTAB, SDTAB, and TABGEN). These are not being included in the COSMIC distribution tapes for reasons of economy.

```
      SUBROUTINE SIGCAL(NG,LG,MG,AG,BG,CG,DG,DGNUL,IGOPTA,FIRSTG,SAVEG,
     1                  EVRG,EVIG,IGERRA,RCONDG,
     2                  OMEGA,RCONDS,ISOPTA,G,SIGMA,U,V,INFO,
     3                  RDUM,CDUM,IDUM,JERR)
      COMPLEX
     + CDUM(*), G(MG,*), U(MG,*), V(LG,*)
      INTEGER
     + JERR, LG, MG, NG, IDUM(*), IGERRA(2), IGOPTA(3), INFO, ISOPTA(3)
      LOGICAL
     + DGNUL, FIRSTG
      REAL
     + OMEGA, RCONDG, RCONDS, AG(*), BG(*), CG(*), DG(*), EVIG(*),
     + EVRG(*), RDUM(*), SAVEG(*), SIGMA(*)
      COMMON /ORACIO/ NIN, NOUT, NERR
      INTEGER
     + NERR, NIN, NOUT
      SAVE /ORACIO/
C     SIGCAL GENERATES A VARIETY OF MATRICES USED FOR THE
C     FREQUENCY DOMAIN ANALYSIS OF MULTIVARIABLE CONTINUOUS OR DISCRETE
C     CONTROL SYSTEMS.  OPTIONS ARE PROVIDED TO COMPUTE SINGULAR VALUES
C     AND VECTORS.
C
C                +----+
C        +       I    I
C      >----O---->I GO I----+--->
C           ^    I    I    I
C         - I    +----+    I
C           I              I
C           +--------------+
C
C     FIRST, SIGCAL REQUIRES THE TRANSFER MATRIX GO EVALUATED AT
C     S = SQRT(-1)*OMEGA (FOR THE CONTINUOUS CASE) OR AT
C     S = EXP(SQRT(-1)*OMEGA) (FOR THE DISCRETE CASE).
C     THE USER HAS THE OPTION OF CALCULATING THIS MATRIX AND PASSING IT
C     TO SIGCAL IN ARRAY G.  IN THE AUTHORS' VIEW, THE "NORMAL"
C     OPERATING MODE OF FREQ WILL BE TO HAVE SIGCAL CALCULATE
```

```
C   GO = CG*((S*(IDENTITY MATRIX) - AG)INVERSE)*BG + DG
C   FROM USER-SUPPLIED LINEAR STATE-SPACE REALIZATION MATRICES
C   (AG, BG, CG, DG).
C
C    THE FOLLOWING MATRICES CAN THEN BE COMPUTED.
C     (1) LOOP-GAIN TRANSFER MATRIX.
C         G = GO
C
C     (2) FEEDBACK TRANSFER MATRIX ( GO SQUARE ).
C         G = GO((I + GO)INVERSE)
C           =  ( I + (GO INVERSE))INVERSE  WHEN GO IS NONSINGULAR
C
C     (3) SENSITIVITY MATRIX ( GO SQUARE ).
C         G =  (I + GO)INVERSE
C
C     (4) INVERSE RETURN DIFFERENCE MATRIX  ( GO SQUARE ).
C         G =  I + (GO INVERSE)   WHEN GO IS NONSINGULAR
C
C     (5) RETURN DIFFERENCE MATRIX  ( GO SQUARE ).
C         G = I + GO
C
C
C    WHEN THE SINGULAR VALUE DECOMPOSITION IS CALCULATED
C
C         G  =  U*(DIAG(SIGMA(I)))*(V CONJUGATE TRANSPOSE)
C
C
C    THE WORKING SYSTEM MATRIX
C
C             ONE MODULE OF THIS CODE, SUBROUTINE SFRMG, ACCEPTS AS
C             INPUT OMEGA, AG, BG, AND CG, AND COMPUTES THE LOOP-GAIN
C             TRANSFER MATRIX GO.  WHEN DG = 0, AN ALTERNATE FORMULA
C             FOR THE FEEDBACK TRANSFER MATRIX (2) IS
C                       CG((SI-AG+BG*CG)INVERSE)BG.
C             THIS CODE USES THE LOOP-GAIN TRANSFER MATRIX MODULE WITH
C             AG REPLACED BY AG-BG*CG TO CALCULATE THE FEEDBACK
C             TRANSFER MATRIX.  ACCORDINGLY, IF ISOPTA(1) = 2 OR 12 AND
C             DG = 0, THE WORKING SYSTEM MATRIX IS AG-BG*CG; OTHERWISE
C             IT IS AG.
C
C
C PARAMETERS
C
C    NG       INTEGER
C             IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C             ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C             USED AS FOLLOWS:
C             ON ENTRY, NUMBER OF STATES
```

```
C            ON RETURN, UNCHANGED
C
C   LG      INTEGER
C            ON ENTRY, NUMBER OF INPUTS
C            ON RETURN, UNCHANGED
C
C    MG      INTEGER
C            ON ENTRY, NUMBER OF OUTPUTS
C            ON RETURN, UNCHANGED
C
C    AG      REAL NGXNG MATRIX STORED AS PACKED ONE-DIMENSIONAL ARRAY
C            IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C            ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C            USED AS FOLLOWS:
C            ON ENTRY, SYSTEM MATRIX
C            ON RETURN, UNCHANGED
C
C    BG      REAL NGXLG MATRIX STORED AS PACKED ONE-DIMENSIONAL ARRAY
C            IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C            ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C            USED AS FOLLOWS:
C            ON ENTRY, INPUT INFLUENCE MATRIX
C            ON RETURN, UNCHANGED
C
C    CG      REAL MGXNG MATRIX STORED AS PACKED ONE-DIMENSIONAL ARRAY
C            IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C            ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C            USED AS FOLLOWS:
C            ON ENTRY, SYSTEM OUTPUT MATRIX
C            ON RETURN, UNCHANGED
C
C    DG      REAL MGXLG MATRIX STORED AS PACKED ONE-DIMENSIONAL ARRAY
C            IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C            ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C            USED AS FOLLOWS:
C            ON ENTRY, IF DGNUL = .FALSE., SYSTEM FEEDFORWARD MATRIX
C                      IF DGNUL = .TRUE., NOT USED, BUT MUST APPEAR AS
C                         AN ARGUMENT OF THE CALLING SEQUENCE
C            ON RETURN, UNCHANGED
C
C    DGNUL   LOGICAL VARIABLE
C            IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C            ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C            USED AS FOLLOWS:
C            ON ENTRY, IF DGNUL = .TRUE., SIGCAL ASSUMES THAT THE
C              SYSTEM FEEDFORWARD MATRIX IS NULL AND IGNORES THE
C              CONTENTS OF DG
C                      IF DGNUL = .FALSE., SIGCAL TAKES THE SYSTEM
```

```
C               FEEDFORWARD MATRIX FROM ARRAY DG
C             ON RETURN, UNCHANGED
C
C     IGOPTA   INTEGER ARRAY OF DIMENSION 3
C             ON ENTRY, CONTROLS OPTIONS AVAILABLE DURING CALCULATION
C               OF THE LOOP-GAIN TRANSFER MATRIX GO
C             ON RETURN, UNCHANGED
C
C        IGOPTA(1)
C             IGOPTA(1) IS IGNORED IF FIRST = .FALSE.
C               WHEN FIRST = .TRUE. AND IGOPTA(1) .GE. 0, THE WORKING
C               SYSTEM MATRIX SUBMITTED TO SFRMG WILL BE BALANCED AND
C               THE EIGENVALUES WILL BE COMPUTED.  IF IGOPTA(1) .LT. 0,
C               THE WORKING SYSTEM MATRIX WILL NOT BE BALANCED AND THE
C               EIGENVALUES WILL NOT BE COMPUTED.
C
C        IGOPTA(2)
C             LET H = SI-A' WHERE S = IS AS ABOVE AND A' IS THE
C               RESULT OF REDUCING THE WORKING SYSTEM MATRIX TO UPPER
C               HESSENBERG FORM.  IF IGOPTA(2) .LT. 0, THE CONDITION
C               NUMBER OF H WILL BE ESTIMATED AND ITS RECIPROCAL
C               RETURNED IN  RCONDG.  IF IGOPTA(2) .GE. 0, THE
C               CONDITION NUMBER OF H WITH RESPECT TO INVERSION WILL
C               NOT BE ESTIMATED.
C
C        IGOPTA(3)
C             IF IGOPTA(3) = 0,  SIGCAL  CALCULATES
C               GO = CG((SI - AG)INVERSE)BG + DG  EVALUATED AT
C               S = SQRT(-1)*OMEGA OR EXP(SQRT(-1)*OMEGA).
C             IF IGOPTA(3) = 1,  SIGCAL ASSUMES THAT THE USER
C               HAS CALCULATED  GO  AND PLACED IT IN ARRAY  G.
C             OTHER VALUES OF IGOPTA(3) GENERATE AN ERROR RETURN.
C
C     FIRSTG   LOGICAL
C             IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C             ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C             USED AS FOLLOWS:
C             ON ENTRY, IF FIRST = .TRUE., SIGCAL STORES INTERMEDIATE
C               RESULTS IN ARRAY SAVEG WHICH CAN BE REUSED WITH NEW
C               OMEGA VALUES.
C                       IF FIRSTG = .FALSE., SIGCAL ASSUMES THAT NG,
C               RG, MG, AG, BG, CG, SAVEG, AND THE WORKING SYSTEM
C               MATRIX (SEE ABOVE) ARE UNCHANGED SINCE IT WAS CALLED
C               WITH FIRSTG = .TRUE. AND RECOVERS INTERMEDIATE VALUES
C               FROM SAVEG INSTEAD OF RECALCULATING THEM.
C             ON RETURN, SIGCAL SETS FIRSTG = .FALSE.
C
C     SAVEG    REAL ARRAY OF DIMENSION AT LEAST NG*(MG+NG+LG)
```

```
C          IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C          ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C          USED AS FOLLOWS:
C          ON ENTRY, IF FIRSTG = .TRUE., NOT USED (NO INPUT DATA
C            REQUIRED, RESERVED FOR OUTPUT STORAGE).
C                    IF FIRSTG = .FALSE., MUST CONTAIN VALUES PLACED
C            THERE BY PRIOR ENTRY INTO SIGCAL WITH FIRSTG = .TRUE.
C          ON RETURN, IF FIRSTG = .TRUE., CONTAINS VALUES WHICH
C            SIGCAL CAN USE ON SUBSEQUENT CALLS WITH FIRSTG =
C            .FALSE.
C                     IF FIRSTG = .FALSE., UNCHANGED.
C
C   EVRG   REAL ARRAY OF DIMENSION AT LEAST NG.
C          IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C          ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C          USED AS FOLLOWS:
C          NOT REQUIRED IF IGOPTA(1) .LT. 0 OR FIRSTG = .FALSE.,
C            BUT STILL MUST APPEAR IN THE CALLING SEQUENCE.
C          ON ENTRY, NOT USED.
C          ON RETURN, IF FIRSTG = .TRUE. AND IGOPTA(1) .GE. 0,
C            EVRG CONTAINS THE REAL PARTS OF THE EIGENVALUES
C            OF THE WORKING SYSTEM MATRIX.  OTHERWISE, EVRG IS
C            UNCHANGED.
C
C   EVIG   REAL ARRAY OF DIMENSION AT LEAST NG.
C          IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C          ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C          USED AS FOLLOWS:
C          ALL REMARKS ABOUT EVRG (PRECEDING) APPLY EXCEPT THAT,
C            IF FIRSTG = .TRUE. AND IGOPTA(1) .GE. 0,
C            EVIG IS SET TO THE IMAGINARY PARTS OF EIGENVALUES
C            CORRESPONDING TO REAL PARTS IN EVRG.
C
C   IGERRA  INTEGER ARRAY OF DIMENSION 2
C          ON ENTRY, NOT USED.
C          ON RETURN, ERROR CODES FOR INDIVIDUAL POTENTIAL PROBLEMS
C            IN THE MODULE THAT CALCULATES THE LOOP-GAIN TRANSFER
C            MATRIX.
C
C      IGERRA(1)
C          ON RETURN, IF FIRSTG = .TRUE. AND IGOPTA(1) .GE. 0 ON
C            ENTRY, THEN IGERRA(1) IS RETURNED AS THE IERR
C            PARAMETER OF THE EISPACK SUBROUTINE HQR -- NORMAL
C            RETURN IS 0.  OTHERWISE, IGERRA(1) IS UNCHANGED.
C
C      IGERRA(2)
C          ON RETURN, IF IGOPTA(2) .GE. 0 ON ENTRY, IGERRA(2) IS
C            RETURNED AS THE INFO PARAMETER OF SUBROUTINE CHEFA.
```

```
C              OTHERWISE, IF 1.0 + RCONDG .EQ. 1.0, (H DEFINED UNDER
C              'IGOPT(2)' ABOVE IS SINGULAR TO WORKING PRECISION),
C              IGERRA(2) IS RETURNED SET TO -1.  NORMAL RETURN IN
C              EITHER CASE IS 0.
C
C     RCONDG   REAL SCALAR
C            IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C            ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C            USED AS FOLLOWS:
C            ON ENTRY, NOT USED.
C            ON RETURN, IF IGOPTA(2) .LT. 0, THE RECIPROCAL OF THE
C              CONDITION NUMBER OF THE H MATRIX DEFINED IN THE
C              DISCUSSION OF IGOPTA(2) ABOVE IS RETURNED IN RCONDG.
C              IF IGOPTA(2) .GE. 0, RCONDG IS UNCHANGED.
C
C     OMEGA    REAL SCALAR
C            ON ENTRY, FREQUENCY (IN THE CONTINUOUS CASE, SHOULD BE
C                      POSITIVE).
C                      NORMALIZED FREQUENCY (IN THE DISCRETE CASE,
C                      SHOULD BE ANGULAR FREQUENCY TIMES SAMPLING
C                      RATE).
C            ON RETURN, UNCHANGED.
C
C     RCONDS   REAL SCALAR
C            ON ENTRY, NOT USED.
C            ON RETURN, IF ISOPTA(1) = 2 OR 12 AND DGNUL = .FALSE., OR
C              IF ISOPTA(1) = 3 OR 13, OR IF ISOPTA(1) = 4 OR 14, G IS
C              CALCULATED FROM GO BY SOLVING THE MATRIX EQUATION
C              (I + GO)G = GO,  (I + GO)G = I,  OR  GO(G - I) = I,
C              RESPECTIVELY, FOR G.  THE SUBROUTINE, CSOLVE, WHICH
C              SOLVES THESE EQUATIONS IS A DRIVER PROGRAM FOR STANDARD
C              LINPACK MODULES OF THE CGE-- FAMILY.  THE ESTIMATED
C              RECIPROCAL CONDITION NUMBER OF THE COEFFICIENT MATRIX
C              CALCULATED BY CGECO IS RETURNED IN RCONDS.  IN THIS
C              CASE, RCONDS IS BETWEEN 0.0 AND 1.0.  IF RCONDS IS SO
C              SMALL THAT 1.0 + RCONDS .EQ. 1.0 TO WITHIN MACHINE
C              PRECISION, THE EQUATION IS TOO ILL-CONDITIONED TO
C              SOLVE.  THIS ERROR CONDITION IS ALSO SIGNALED BY
C              SETTING JERR = 11.
C                IF CSOLVE IS NOT USED, RCONDS IS SET TO 2.0.
C
C     ISOPTA   INTEGER ARRAY OF DIMENSION 3
C            ON ENTRY, TRANSMITS USER SELECTED OPTIONS TO SIGCAL --
C              WHICH G MATRIX TO COMPUTE AND WHETHER TO USE
C              CONTINUOUS OR DISCRETE FORMULATION; WHETHER TO COMPUTE
C              SINGULAR VALUES OR SINGULAR VECTORS; AND WHETHER TO
C              USE THE BUILT-IN PRINTOUT FEATURE.
C            ON RETURN, UNCHANGED.
```

```
C
C       ISOPTA(1)
C             .EQ. 1, CONTINUOUS LOOP-GAIN TRANSFER MATRIX RETURNED
C                     IN G.
C             .EQ. 2, CONTINUOUS FEEDBACK TRANSFER MATRIX RETURNED IN
C                     G.
C             .EQ. 3, CONTINUOUS SENSITIVITY MATRIX RETURNED IN G.
C             .EQ. 4, CONTINUOUS INVERSE RETURN DIFFERENCE MATRIX
C                     RETURNED IN G.
C             .EQ. 5, CONTINUOUS RETURN DIFFERENCE MATRIX RETURNED IN
C                     G.
C             .EQ. 11, DISCRETE LOOP-GAIN TRANSFER MATRIX RETURNED
C                      IN G.
C             .EQ. 12, DISCRETE FEEDBACK TRANSFER MATRIX RETURNED IN
C                      G.
C             .EQ. 13, DISCRETE SENSITIVITY MATRIX RETURNED IN G.
C             .EQ. 14, DISCRETE INVERSE RETURN DIFFERENCE MATRIX
C                      RETURNED IN G.
C             .EQ. 15, DISCRETE RETURN DIFFERENCE MATRIX RETURNED IN
C                      G.
C
C       ISOPTA(2)
C             .EQ. 1, DO NOT COMPUTE SINGULAR VALUES OR VECTORS OF G.
C                     RETURN AFTER COMPUTING G.
C             .EQ. 2, COMPUTE SINGULAR VALUES BUT NO SINGULAR
C                     VECTORS OF G.
C             .EQ. 3, COMPUTE BOTH SINGULAR VALUES AND VECTORS OF G.
C
C       ISOPTA(3)
C             .NE. 0, PRINT INPUT, PROBLEM STATEMENT, AND RESULTS OF
C                     CALCULATIONS AS THEY PROCEED.
C             .EQ. 0, NO PRINTING EXCEPT FOR ERROR MESSAGES.
C
C   G         COMPLEX MGXLG MATRIX STORED AS A PACKED ONE-DIMENSIONAL
C               COMPLEX ARRAY
C             ON ENTRY, IF IGOPTA(3) = 0, NOT USED.
C                       IF IGOPTA(3) = 1, THE USER MUST PUT THE  GO
C                       MATRIX, EVALUATED AT  S = SQRT(-1)*OMEGA
C                       OR EXP(SQRT(-1)*OMEGA), IN THIS ARRAY.
C             ON RETURN, CONTAINS THE G-MATRIX SELECTED BY ISOPTA(1).
C
C   SIGMA     REAL ARRAY OF DIMENSION AT LEAST MIN(MG,LG).
C             NOT REQUIRED IF ISOPTA(2) = 1, BUT STILL MUST APPEAR AS
C               AN ARGUMENT OF CALLING SEQUENCE.
C             ON ENTRY, NOT USED.
C             ON NORMAL RETURN, SIGMA CONTAINS THE SINGULAR VALUES OF
C               G ARRANGED IN DESCENDING ORDER OF MAGNITUDE.
C             ON ERROR RETURN, FIRST MIN(M,R)-INFO ENTRIES OF SIGMA ARE
```

```
C                SINGULAR VALUES, REMAINING ENTRIES ARE UNDEFINED.
C
C     U        COMPLEX  MG X MIN(LG,MG)  MATRIX STORED AS A PACKED
C                ONE-DIMENSIONAL COMPLEX ARRAY
C              NOT REQUIRED IF ISOPTA(2) = 1 OR 2, BUT A DUMMY
C                ARGUMENT MUST APPEAR IN THIS POSITION OF THE CALLING
C                SEQUENCE.
C              IF ISOPTA(2) = 3, U SERVES THE FOLLOWING FUNCTION.
C              ON ENTRY, NOT USED.
C              ON NORMAL RETURN, CONTAINS THE LEFT SINGULAR VECTORS
C                OF G.
C              ON ERROR RETURN, SEE THE DISCUSSION OF INFO BELOW.
C
C     V        COMPLEX  LG X MIN(LG,MG)  MATRIX STORED AS A PACKED
C                ONE-DIMENSIONAL COMPLEX ARRAY
C              NOT REQUIRED IF ISOPTA(2) = 1 OR 2, BUT A DUMMY
C                ARGUMENT MUST APPEAR IN THIS POSITION OF THE CALLING
C                SEQUENCE.
C              IF ISOPTA(2) = 3, V SERVES THE FOLLOWING FUNCTION.
C              ON ENTRY, NOT USED.
C              ON NORMAL RETURN, CONTAINS THE RIGHT SINGULAR VECTORS
C                OF G.
C              ON ERROR RETURN, SEE THE DISCUSSION OF INFO BELOW.
C
C     INFO     INTEGER
C              ON ENTRY, NOT USED.
C              ON RETURN, ERROR CODE FOR THE G-MATRIX SINGULAR VALUES
C                CALCULATION.  IF ISOPTA(2) = 2 OR 3, INFO IS SET TO THE
C                "INFO" PARAMETER RETURNED FROM SINGULAR VALUE
C                DECOMPOSITION SUBROUTINE CSVDC.  OTHERWISE INFO IS SET
C                TO 0.  NORMAL RETURN IS 0.  IN THE EVENT OF AN ERROR
C                RETURN FROM CSVDC, REFER TO THE DOCUMENTATION OF
C                LINPACK SUBROUTINE CSVDC FOR THE MEANING OF THE
C                CONTENTS OF MATRICES U AND V.  IN THAT DOCUMENTATION,
C                REFERENCE IS MADE TO COMPLEX VECTORS S AND E. IN THE
C                EVENT OF AN ERROR RETURN, THE CONTENTS OF S AND E ARE
C                PRESERVED AS, RESPECTIVELY, THE FIRST MIN(MG+1,LG) AND
C                THE NEXT MIN(MG,LG) ENTRIES IN ARRAY CDUM (SEE BELOW).
C
C     RDUM     REAL SCRATCH ARRAY OF DIMENSION AT LEAST NG.
C
C     CDUM     COMPLEX SCRATCH ARRAY.
C               DIMENSIONAL NEEDS OF CDUM VARY WITH APPLICATION.  AN
C                 UPPER BOUND ON SPACE NEEDED IS 2*K*(K+1)+1 LOCATIONS
C                 OF TYPE COMPLEX, WHERE K = MAX(MG,NG,LG).  EXACT
C                 STORAGE NEEDED BY CDUM IS CALCULATED AS FOLLOWS.  LET
C                  I1 = NG*(NG+LG+1).  IF ISOPTA(1) = 2 OR 12 AND DGNUL =
C                  .FALSE., OR IF ISOPTA(1) = 3, 4, 13, OR 14, LET I2 =
C                MG*(2*MG+1); OTHERWISE, I2 = 0.  IF ISOPTA(2) IS 2
C                OR 3 AND MG .LT. LG, LET I3 = (MG+1)*(LG+2)-1;
C                OTHERWISE, I3 = 0.  IF ISOPTA(2) IS 2 OR 3 AND MG
C                .GE. LG, LET I4 = LG*(MG+2)+MG; OTHERWISE I4 = 0.
C                THEN CDUM NEEDS MAX(I1,I2,I3,I4) TYPE COMPLEX
C                LOCATIONS.
C
C     IDUM      INTEGER SCRATCH ARRAY OF DIMENSION AT LEAST MAX(MG,NG)
C
C     JERR      INTEGER
C               IF NO ERROR IS DETECTED BY SIGCAL, JERR IS SET TO 0.
C               IF AN ERROR IS DETECTED, JERR IS SET TO THE FIRST
C                 APPLICABLE OF THE FOLLOWING ERROR CODES.
C                   1  MATRIX DIMENSIONS NONPOSITIVE OR INCONSISTENT
C                      WITH CALCULATION CHOSEN BY ISOPTA(1).
C                   2  HQR ERROR OCCURRED, SEE IGERRA(1) ABOVE.
C                   3  INFO PARAMETER OF CHEFA INDICATES AN ERROR,
C                      SEE DISCUSSION OF IGERRA(2) ABOVE.
C                   4  1. .EQ. 1. + RCONDG, H IS (NEARLY) SINGULAR,
C                      SEE DISCUSSION OF RCONDG ABOVE.
C                   5  IGOPTA(3) WAS NEITHER 0 NOR 1.
C                  11  CSOLVE ERROR OCCURRED, SEE RCONDS ABOVE.
C                  12  CSVDC ERROR OCCURRED, SEE INFO ABOVE.
```

```
      SUBROUTINE SIGDYN(NG,LG,MG,AG,BG,CG,DG,DGNUL,IGOPTA,FIRSTG,SAVEG,
     1                  EVRG,EVIG,GO,IGERRA,RCONDG,
     2                  NK,LK,MK,AK,BK,CK,DK,DKNUL,IKOPTA,FIRSTK,SAVEK,
     3                  EVRK,EVIK,KO,IKERRA,RCONDK,
     4                  OMEGA,RCONDS,ISOPTA,G,SIGMA,U,V,INFO,
     5                  RDUM,CDUM,IDUM,JERR)
      COMPLEX
     + CDUM(*), G(*), GO(MG,*), KO(MK,*), U(*), V(*)
      INTEGER
     + JERR, MG, MK, NG, NK, LG, LK, IDUM(*), IGERRA(2), IGOPTA(3),
     + IKERRA(2), IKOPTA(3), INFO, ISOPTA(4)
      LOGICAL
     + DGNUL, DKNUL, FIRSTG, FIRSTK
      REAL
     + OMEGA, RCONDG, RCONDK, RCONDS, AG(*), AK(*), BG(*), BK(*), CG(*),
     + CK(*), DG(*), DK(*), EVIG(*), EVIK(*), EVRG(*), EVRK(*), RDUM(*),
     + SAVEG(*), SAVEK(*), SIGMA(*)
      COMMON /ORACIO/ NIN, NOUT, NERR
      INTEGER
     + NERR, NIN, NOUT
      SAVE /ORACIO/
C
C     SIGDYN GENERATES A VARIETY OF MATRICES USED FOR THE
C     FREQUENCY DOMAIN ANALYSIS OF MULTIVARIABLE CONTINUOUS OR DISCRETE
C     CONTROL SYSTEMS.
C
C                +----+     +----+
C        +       I    I     I    I
C     >----O---->I KO I---->I GO I ----+--->
C          ^     I    I     I    I     I
C        - I     +----+     +----+     I
C          I                           I
C          +---------------------------+
C
C     SIGDYN  FIRST DETERMINES IF THE USER HAS SUPPLIED EITHER OR BOTH
C     OF THE SYSTEM AND COMPENSATOR DYNAMICS MATRICES,  GO  AND  KO.
C     ANY WHICH HAS NOT BEEN USER-SUPPLIED IS CALCULATED USING
C     THE FORMULAS
C        GO = CG((SI - AG)INVERSE)BG + DG,  AND
C        KO = CK((SI - AK)INVERSE)BK + DK
C     EVALUATED AT S = SQRT(-1)*OMEGA (CONTINUOUS CASE) OR AT
C     S = EXP(SQRT(-1)*OMEGA) (DISCRETE CASE).
C
C     IN THE FOLLOWING,  GL  STANDS FOR THE USER'S CHOICE OF ONE OF
C     EIGHT MATRICES COMPUTED FROM  GO  AND  KO  BY  GL = G' * K' OR
C     K' * G'  WHERE  G' = GO OR (GO)INVERSE  AND  K' = KO OR
C     (KO)INVERSE.
C
```

```
C    THE FOLLOWING MATRICES CAN THEN BE COMPUTED.
C     (1) LOOP-GAIN TRANSFER MATRIX.
C         G = GL
C
C     (2) FEEDBACK TRANSFER MATRIX ( GL SQUARE ).
C         G = GL((I + GL)INVERSE)
C           =  ( I + (GL INVERSE))INVERSE   WHEN GL IS NONSINGULAR
C
C     (3) SENSITIVITY MATRIX ( GL SQUARE ).
C         G =  (I + GL)INVERSE
C
C     (4) INVERSE RETURN DIFFERENCE MATRIX  ( GL SQUARE ).
C         G =  I + (GL INVERSE)   WHEN GL IS NONSINGULAR
C
C     (5) RETURN DIFFERENCE MATRIX  ( GL SQUARE ).
C         G = I + GL
C
C    OPTIONS ARE PROVIDED TO COMPUTE SINGULAR VALUES AND VECTORS.
C
C    WHEN THE SINGULAR VALUE DECOMPOSITION IS CALCULATED
C
C        G  =  U*(DIAG(SIGMA(I)))*(V CONJUGATE TRANSPOSE)
C
C
C PARAMETERS.  ***NOTE***  THE FIVE INTEGER VARIABLES IKG, IGINV,
C    IKINV, M, AND L ARE DEFINED IN TERMS OF THE PARAMETERS ISOPTA, MG,
C    LG, MK, AND LK FROM THE CALLING SEQUENCE OF  SIGDYN  AND ARE USED
C    THROUGHOUT THE FOLLOWING DISCUSSION.  THEY ARE DEFINED AS FOLLOWS:
C      SINCE ISOPTA(4) IS RESTRICTED TO BE BETWEEN 0 AND 7 INCLUSIVE,
C      IKG, IGINV, AND IKINV ARE UNIQUELY DEFINED BY THE REQUIREMENTS
C      THAT EACH BE EITHER 0 OR 1 AND
C        ISOPTA(4) = IKG + 2*IGINV + 4*IKINV.
C      IF IKG = 0, THEN  M  IS DEFINED TO BE  MG  AND  L  TO BE  LK.
C      IF IKG = 1, THEN  M  IS DEFINED TO BE  MK  AND  L  TO BE  LG.
C
C    NG        INTEGER
C              IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C              ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C              USED AS FOLLOWS:
C              ON ENTRY, NUMBER OF SYSTEM STATES
C              ON RETURN, UNCHANGED
C
C    LG        INTEGER
C              ON ENTRY, NUMBER OF SYSTEM INPUTS
C              ON RETURN, UNCHANGED
C
```

```
C    MG        INTEGER
C              ON ENTRY, NUMBER OF SYSTEM OUTPUTS
C              ON RETURN, UNCHANGED
C
C    AG        REAL NGXNG MATRIX STORED AS PACKED ONE-DIMENSIONAL ARRAY
C              IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C              ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C              USED AS FOLLOWS:
C              ON ENTRY, SYSTEM MATRIX
C              ON RETURN, UNCHANGED
C
C    BG        REAL NGXLG MATRIX STORED AS PACKED ONE-DIMENSIONAL ARRAY
C              IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C              ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C              USED AS FOLLOWS:
C              ON ENTRY, INPUT INFLUENCE MATRIX
C              ON RETURN, UNCHANGED
C
C    CG        REAL MGXNG MATRIX STORED AS PACKED ONE-DIMENSIONAL ARRAY
C              IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C              ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C              USED AS FOLLOWS:
C              ON ENTRY, SYSTEM OUTPUT MATRIX
C              ON RETURN, UNCHANGED
C
C    DG        REAL MGXLG MATRIX STORED AS PACKED ONE-DIMENSIONAL ARRAY
C              IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C              ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C              USED AS FOLLOWS:
C              ON ENTRY, IF DGNUL = .FALSE., SYSTEM FEEDFORWARD MATRIX
C                        IF DGNUL = .TRUE., NOT USED, BUT MUST APPEAR AS
C                           AN ARGUMENT OF THE CALLING SEQUENCE
C              ON RETURN, UNCHANGED
C
C    DGNUL     LOGICAL VARIABLE
C              IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C              ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C              USED AS FOLLOWS:
C              ON ENTRY, IF DGNUL = .TRUE., SIGDYN ASSUMES THAT THE
C                SYSTEM FEEDFORWARD MATRIX IS NULL AND IGNORES THE
C                CONTENTS OF DG
C                        IF DGNUL = .FALSE., SIGDYN TAKES THE SYSTEM
C                FEEDFORWARD MATRIX FROM ARRAY DG
C              ON RETURN, UNCHANGED
C
C    IGOPTA    INTEGER ARRAY OF DIMENSION 3
C              ON ENTRY, CONTROLS OPTIONS AVAILABLE DURING CALCULATION
C                OF THE SYSTEM DYNAMICS MATRIX GO
C              ON RETURN, UNCHANGED
C
```

```
C       IGOPTA(1)
C            IGOPTA(1) IS IGNORED IF FIRSTG = .FALSE.
C              WHEN FIRSTG = .TRUE. AND IGOPTA(1) .GE. 0, AG WILL BE
C              BALANCED AND ITS EIGENVALUES WILL BE COMPUTED.  IF
C              IGOPTA(1) .LT. 0, AG WILL NOT BE BALANCED AND ITS
C              EIGENVALUES WILL NOT BE COMPUTED.
C
C       IGOPTA(2)
C            LET H = SI-A' WHERE S = IS AS ABOVE AND A' IS THE
C              RESULT OF REDUCING AG TO UPPER HESSENBERG FORM.  IF
C              IGOPTA(2) .LT. 0, THE CONDITION NUMBER OF H WITH
C              RESPECT TO INVERSION WILL BE ESTIMATED AND ITS
C              RECIPROCAL RETURNED IN  RCONDG.  IF IGOPTA(2) .GE. 0,
C              THE CONDITION NUMBER OF H WITH RESPECT TO INVERSION
C              WILL NOT BE ESTIMATED.
C
C       IGOPTA(3)
C            IF IGOPTA(3) = 0,  SIGCAL  CALCULATES
C              GO = C((SI - A)INVERSE)B + D  EVALUATED AT
C              S = SQRT(-1)*OMEGA OR EXP(SQRT(-1)*OMEGA).
C            IF IGOPTA(3) = 1,  SIGCAL ASSUMES THAT THE USER
C              HAS CALCULATED  GO  (Q. V.).
C            OTHER VALUES OF IGOPTA(3) GENERATE AN ERROR RETURN.
C
C   FIRSTG   LOGICAL
C            IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C            ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C            USED AS FOLLOWS:
C            ON ENTRY, IF FIRSTG = .TRUE., SIGDYN STORES INTERMEDIATE
C              RESULTS IN ARRAY SAVEG WHICH CAN BE REUSED WITH NEW
C              OMEGA VALUES.
C                      IF FIRSTG = .FALSE., SIGDYN ASSUMES THAT NG,
C              LG, MG, AG, BG, CG, AND SAVEG ARE UNCHANGED SINCE IT
C              WAS CALLED WITH FIRSTG = .TRUE. AND RECOVERS
C              INTERMEDIATE VALUES FROM SAVEG INSTEAD OF
C              RECALCULATING THEM.
C            ON RETURN, SIGDYN SETS FIRSTG = .FALSE.
C
C   SAVEG    REAL ARRAY OF DIMENSION AT LEAST NG*(MG+NG+LG)
C            IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C            ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C            USED AS FOLLOWS:
C            ON ENTRY, IF FIRSTG = .TRUE., NOT USED (NO INPUT DATA
C              REQUIRED, RESERVED FOR OUTPUT STORAGE).
C                      IF FIRSTG = .FALSE., MUST CONTAIN VALUES PLACED
C              THERE BY PRIOR ENTRY INTO SIGDYN WITH FIRSTG = .TRUE.
C            ON RETURN, IF FIRSTG = .TRUE., CONTAINS VALUES WHICH
C              SIGDYN CAN USE ON SUBSEQUENT CALLS WITH FIRSTG =
C               .FALSE.
C                       IF FIRSTG = .FALSE., UNCHANGED.
C
C    EVRG      REAL ARRAY OF DIMENSION AT LEAST NG
C              IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C              ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C              USED AS FOLLOWS:
C              NOT REQUIRED IF IGOPTA(1) .LT. 0 OR FIRSTG = .FALSE.,
C                BUT STILL MUST APPEAR IN THE CALLING SEQUENCE.
C              ON ENTRY, NOT USED.
C              ON RETURN, IF FIRSTG = .TRUE. AND IGOPTA(1) .GE. 0,
C                EVRG CONTAINS THE REAL PARTS OF THE EIGENVALUES
C                OF AG.  OTHERWISE, EVRG IS UNCHANGED.
C
C    EVIG      REAL ARRAY OF DIMENSION AT LEAST NG
C              IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C              ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C              USED AS FOLLOWS:
C              ALL REMARKS ABOUT EVRG (PRECEDING) APPLY EXCEPT THAT,
C                IF FIRSTG = .TRUE. AND IGOPTA(1) .GE. 0,
C                EVIG IS SET TO THE IMAGINARY PARTS OF EIGENVALUES
C                CORRESPONDING TO REAL PARTS IN EVRG.
C
C    GO        COMPLEX ARRAY OF DIMENSION AT LEAST MG*LG
C              ON ENTRY, IF IGOPTA(3) = 0, NOT USED.
C                        IF IGOPTA(3) = 1, CONTAINS THE SYSTEM DYNAMICS
C                        MATRIX EVALUATED AT  S = SQRT(-1)*OMEGA OR
C                        S = EXP(SQRT(-1)*OMEGA).
C              ON RETURN, IF IGOPTA(3) = 0, CONTAINS
C                        CG((SI-AG)INVERSE)BG + DG
C                        FOR  S = SQRT(-1)*OMEGA OR
C                             S = EXP(SQRT(-1)*OMEGA).
C                         IF IGOPTA(3) = 1, UNCHANGED.
C
C    IGERRA    INTEGER ARRAY OF DIMENSION 2
C              ON ENTRY, NOT USED.
C              ON RETURN, ERROR CODES FOR INDIVIDUAL POTENTIAL PROBLEMS
C                IN THE MODULE THAT CALCULATES GO.
C
C        IGERRA(1)
C              ON RETURN, IF FIRSTG = .TRUE. AND IGOPTA(1) .GE. 0 ON
C                ENTRY, THEN IGERRA(1) IS RETURNED AS THE IERR
C                PARAMETER OF THE EISPACK SUBROUTINE HQR -- NORMAL
C                RETURN IS 0.  OTHERWISE, IGERRA(1) IS UNCHANGED.
C
C        IGERRA(2)
C              ON RETURN, IF IGOPTA(2) .GE. 0 ON ENTRY, IGERRA(2) IS
C                RETURNED AS THE INFO PARAMETER OF SUBROUTINE ZHEFA.
```

```
C              OTHERWISE, IF 1.0 + RCONDG .EQ. 1.0, (H DEFINED UNDER
C              'IGOPT(2)' ABOVE IS SINGULAR TO WORKING PRECISION),
C              IGERRA(2) IS RETURNED SET TO -1.  NORMAL RETURN IN
C              EITHER CASE IS 0.
C
C     RCONDG   REAL SCALAR
C              IF IGOPTA(3) .NE. 0, UNUSED, BUT MUST APPEAR AS AN
C              ARGUMENT IN THE CALLING SEQUENCE.  IF IGOPTA(3) = 0,
C              USED AS FOLLOWS:
C              ON ENTRY, NOT USED.
C              ON RETURN, IF IGOPTA(2) .LT. 0, THE RECIPROCAL OF THE
C                CONDITION NUMBER OF THE H MATRIX DEFINED IN THE
C                DISCUSSION OF IGOPTA(2) ABOVE IS RETURNED IN RCONDG.
C                IF IGOPTA(2) .GE. 0, RCONDG IS UNCHANGED.
C
C     NK,LK,MK,AK,BK,CK,DK,DKNUL,IKOPTA,FIRSTK,SAVEK,
C     EVRK,EVIK,KO,IKERRA,RCONDK
C
C              USED FOR THE DYNAMIC COMPENSATOR IN EXACTLY THE SAME WAY
C                AS
C                       NG,LG,MG,AG,BG,CG,DG,DGNUL,IGOPTA,FIRSTG,SAVEG,
C                       EVRG,EVIG,GO,IGERRA,RCONDG
C                RESPECTIVELY, ARE USED FOR THE BASIC SYSTEM.
C
C     OMEGA    REAL SCALAR
C              ON ENTRY, FREQUENCY (IN THE CONTINUOUS CASE, SHOULD BE
C                         POSITIVE).
C                         NORMALIZED FREQUENCY (IN THE DISCRETE CASE,
C                         SHOULD BE ANGULAR FREQUENCY TIMES SAMPLING
C                         RATE).
C              ON RETURN, UNCHANGED.
C
C     RCONDS   REAL SCALAR
C              ON ENTRY, NOT USED.
C              ON RETURN, IF ISOPTA(4) = 2, 3, 4, 5, 6, OR 7, GL IS
C                CALCULATED BY SOLVING THE EQUATION  GO*GL = KO,
C                GO'*GL' = KO',  KO'*GL' = GO',  KO*GL = GO,
C                (KO*GO)*GL = I,  OR (KO*GO)*GL = I,  RESPECTIVELY,
C                FOR GL.  IF ISOPTA(1) = 2 OR 12; OR 3 OR 13; OR 4 OR
C                14; G IS CALCULATED FROM GL BY SOLVING THE MATRIX
C                EQUATION  (I + GL)G = GL,  (I + GL)G = I,  OR
C                GL(G - I) = I, RESPECTIVELY, FOR G.  THE SUBROUTINE,
C                CSOLVE, WHICH SOLVES THESE EQUATIONS IS A DRIVER
C                PROGRAM FOR STANDARD LINPACK MODULES OF THE CGE--
C                FAMILY.  EACH CALL TO CSOLVE RESULTS IN AN ESTIMATE OF
C                THE RECIPROCAL OF THE CONDITION NUMBER OF THE
C                COEFFICIENT MATRIX BEING CALCULATED BY CGECO.  THE
C                MINIMUM OF THESE OVER ALL CALLS TO CSOLVE IS RETURNED
```

```
C               IN RCONDS.  IN THIS CASE, RCONDS IS BETWEEN 0.0 AND
C               1.0.  IF RCONDS IS SO SMALL THAT 1.0 + RCONDS .EQ. 1.0
C               TO WITHIN MACHINE PRECISION, THE EQUATION IS TOO
C               ILL-CONDITIONED TO SOLVE.  THIS ERROR CONDITION IS ALSO
C               SIGNALED BY SETTING JERR = 10 OR 11.
C                 IF CSOLVE IS NOT USED, RCONDS IS SET TO 2.0.
C
C     ISOPTA   INTEGER ARRAY OF DIMENSION 4
C             ON ENTRY, TRANSMITS USER-SELECTED OPTIONS TO SIGDYN --
C               WHICH G MATRIX TO COMPUTE; WHETHER TO COMPUTE
C               SINGULAR VALUES OR SINGULAR VECTORS; WHETHER TO USE
C               THE BUILT-IN PRINTOUT FEATURE; AND WHAT GL IS.
C             ON RETURN, UNCHANGED.
C
C         ISOPTA(1)
C              .EQ. 1, CONTINUOUS LOOP-GAIN TRANSFER MATRIX RETURNED
C                      IN G.
C              .EQ. 2, CONTINUOUS FEEDBACK TRANSFER MATRIX RETURNED IN
C                      G.
C              .EQ. 3, CONTINUOUS SENSITIVITY MATRIX RETURNED IN G.
C              .EQ. 4, CONTINUOUS INVERSE RETURN DIFFERENCE MATRIX
C                      RETURNED IN G.
C              .EQ. 5, CONTINUOUS RETURN DIFFERENCE MATRIX RETURNED IN
C                      G.
C              .EQ. 11, DISCRETE LOOP-GAIN TRANSFER MATRIX RETURNED
C                       IN G.
C              .EQ. 12, DISCRETE FEEDBACK TRANSFER MATRIX RETURNED IN
C                       G.
C              .EQ. 13, DISCRETE SENSITIVITY MATRIX RETURNED IN G.
C              .EQ. 14, DISCRETE INVERSE RETURN DIFFERENCE MATRIX
C                       RETURNED IN G.
C              .EQ. 15, DISCRETE RETURN DIFFERENCE MATRIX RETURNED IN
C                       G.
C         ISOPTA(2)
C              .EQ. 1, DO NOT COMPUTE SINGULAR VALUES OR VECTORS OF G.
C                      RETURN AFTER COMPUTING G.
C              .EQ. 2, COMPUTE SINGULAR VALUES BUT NO SINGULAR
C                      VECTORS OF G.
C              .EQ. 3, COMPUTE BOTH SINGULAR VALUES AND VECTORS OF G.
C
C         ISOPTA(3)
C              .NE. 0, PRINT INPUT, PROBLEM STATEMENT, AND RESULTS OF
C                      CALCULATIONS AS THEY PROCEED.
C              .EQ. 0, NO PRINTING EXCEPT FOR ERROR MESSAGES.
C
C         ISOPTA(4).  MUST BE BETWEEN 0 AND 7.  DETERMINES HOW  GL  WILL
C                     BE CALCULATED FROM GO AND KO.
C             WITH IKG, IGINV, AND IKINV DEFINED AS IN THE NOTE ABOVE,
C              SET  GL = G' * K' IF IKG = 0,
C                   GL = K' * G' IF IKG = 1;
C              WHERE  G' = GO          IF IGINV = 0,
C                     G' = (GO)INVERSE IF IGINV = 1,
C                     K' = KO          IF IKINV = 0, AND
C                     K' = (KO)INVERSE IF IKINV = 1.
C
C    G        COMPLEX MXR MATRIX STORED AS A PACKED ONE-DIMENSIONAL
C               COMPLEX ARRAY
C             ON ENTRY, NOT USED.
C             ON RETURN, CONTAINS THE G-MATRIX SELECTED BY ISOPTA(1).
C
C    SIGMA    REAL ARRAY OF DIMENSION AT LEAST MIN(M,R).
C             NOT REQUIRED IF ISOPTA(2) = 1, BUT STILL MUST APPEAR AS
C               AN ARGUMENT OF CALLING SEQUENCE.
C             ON ENTRY, NOT USED.
C             ON NORMAL RETURN, SIGMA CONTAINS THE SINGULAR VALUES OF
C               G ARRANGED IN DESCENDING ORDER OF MAGNITUDE.
C             ON ERROR RETURN, FIRSTG MIN(M,R)-INFO ENTRIES OF SIGMA
C               ARE SINGULAR VALUES; REMAINING ENTRIES ARE UNDEFINED.
C
C    U        COMPLEX  M X MIN(R,M)  MATRIX STORED AS A PACKED
C               ONE-DIMENSIONAL COMPLEX ARRAY
C             NOT REQUIRED IF ISOPTA(2) = 1 OR 2, BUT A DUMMY
C               ARGUMENT MUST APPEAR IN THIS POSITION OF THE CALLING
C               SEQUENCE.
C             IF ISOPTA(2) = 3, U SERVES THE FOLLOWING FUNCTION.
C             ON ENTRY, NOT USED.
C             ON NORMAL RETURN, CONTAINS THE LEFT SINGULAR VECTORS
C               OF G.
C             ON ERROR RETURN, SEE THE DISCUSSION OF INFO BELOW.
C
C    V        COMPLEX  L X MIN(R,M)  MATRIX STORED AS A PACKED
C               ONE-DIMENSIONAL COMPLEX ARRAY
C             NOT REQUIRED IF ISOPTA(2) = 1 OR 2, BUT A DUMMY
C               ARGUMENT MUST APPEAR IN THIS POSITION OF THE CALLING
C               SEQUENCE.
C             IF ISOPTA(2) = 3, V SERVES THE FOLLOWING FUNCTION.
C             ON ENTRY, NOT USED.
C             ON NORMAL RETURN, CONTAINS THE RIGHT SINGULAR VECTORS
C               OF G.
C             ON ERROR RETURN, SEE THE DISCUSSION OF INFO BELOW.
C
C    INFO     INTEGER
C             ON ENTRY, NOT USED.
C             ON RETURN, ERROR CODE FOR THE G-MATRIX SINGULAR VALUES
C               CALCULATION.  IF ISOPTA(2) = 2 OR 3, INFO IS SET TO THE
C               "INFO" PARAMETER RETURNED FROM SINGULAR VALUE
```

```
C             DECOMPOSITION SUBROUTINE CSVDC.  OTHERWISE INFO IS SET
C             TO 0.  NORMAL RETURN IS 0.  IN THE EVENT OF AN ERROR
C             RETURN FROM CSVDC, REFER TO THE DOCUMENTATION OF
C             LINPACK SUBROUTINE CSVDC FOR THE MEANING OF THE
C             CONTENTS OF MATRICES U AND V.  IN THAT DOCUMENTATION,
C             REFERENCE IS MADE TO COMPLEX VECTORS S AND E. IN THE
C             EVENT OF AN ERROR RETURN, THE CONTENTS OF S AND E ARE
C             PRESERVED AS, RESPECTIVELY, THE FIRST MIN(M+1,L) AND
C             THE NEXT MIN(M,L) ENTRIES IN ARRAY CDUM (SEE BELOW).
C
C    RDUM     REAL SCRATCH ARRAY OF DIMENSION AT LEAST MAX(NG,NK).
C
C    CDUM     COMPLEX SCRATCH ARRAY.
C             DIMENSIONAL NEEDS OF CDUM VARY WITH APPLICATION.  AN
C               UPPER BOUND ON SPACE NEEDED IS J*(3*J+2) LOCATIONS OF
C               TYPE COMPLEX, WHERE J = MAX(MG,NG,LG,MK,NK,LK).  EXACT
C               STORAGE NEEDED BY CDUM IS CALCULATED AS FOLLOWS.
C               LET I1 = NG*(NG+LG+1). LET I2 = NK*(NK+LK+1).
C               IF ISOPTA(1) IS 2, 3, 4, 12, 13, OR 14, LET I3 =
C               M*(2*M+1); OTHERWISE I3 = 0.  IF ISOPTA(2) IS 2 OR 3
C               AND M .LT. L, LET I4 = (M+1)*(L+2)-1; OTHERWISE, I4 =
C               0.  IF ISOPTA(2) IS 2 OR 3 AND M .GE. L, LET I5 =
C               L*(M+2)+M; OTHERWISE I5 = 0.  DETERMINE I6 FROM THE
C               FOLLOWING TABLE:
C
C                     ISOPTA(4)              I6
C                     =========          ===========
C                      0 OR 1                0
C                      2 OR 5              M*(M+1)
C                      3 OR 4            L*(2*L+M+1)
C                      6 OR 7             M*(2*M+1)
C
C               THEN CDUM NEEDS MAX(I1,I2,I3,I4,I5,I6) TYPE COMPLEX
C               LOCATIONS.
C
C    IDUM     INTEGER SCRATCH ARRAY OF DIMENSION AT LEAST MAX(M,NG,NK)
C               UNLESS ISOPTA(4) = 6 OR 7.  THEN IDUM MUST HAVE AT
C               LEAST MAX(M,L,NG,NK) LOCATIONS.
C
C    JERR     INTEGER
C             IF NO ERROR IS DETECTED BY SIGDYN, JERR IS SET TO 0.
C             IF AN ERROR IS DETECTED, JERR IS SET TO THE FIRST
C               APPLICABLE OF THE FOLLOWING ERROR CODES.
C                  1  MATRIX DIMENSIONS NONPOSITIVE OR INCONSISTENT
C                     WITH CALCULATION CHOSEN BY ISOPTA(1) AND/OR
C                     ISOPTA(4), OR ISOPTA(4) OUT OF RANGE.
C                  2  HQR ERROR OCCURRED IN GO CALCULATION; SEE
C                     IGERRA(1) ABOVE.
```

```
C                       3  INFO PARAMETER OF ZHEFA INDICATES AN ERROR IN
C                          GO CALCULATION; SEE DISCUSSION OF IGERRA(2).
C                       4  1. .EQ. 1. + RCONDG; H IS (NEARLY) SINGULAR;
C                          SEE DISCUSSION OF RCONDG ABOVE.
C                       5  IGOPTA(3) WAS NEITHER 0 NOR 1.
C                       6  HQR ERROR OCCURRED IN KO CALCULATION, LIKE
C                          JERR = 2 FOR GO CALCULATION.
C                       7  INFO PARAMETER OF ZHEFA INDICATES AN ERROR IN
C                          KO CALCULATION, LIKE JERR = 3 FOR GO CASE.
C                       8  1. .EQ. 1. + RCONDK, LIKE JERR = 4 FOR GO CASE.
C                       9  IKOPTA(3) WAS NEITHER 0 NOR 1.
C                      10  CSOLVE ERROR OCCURRED IN CALCULATING GL FROM GO
C                          AND KO; SEE RCONDS ABOVE.
C                      11  CSOLVE ERROR OCCURRED IN CALCULATING G FROM GL;
C                          SEE RCONDS ABOVE.
C                      12  CSVDC ERROR OCCURRED; SEE INFO ABOVE.
C                      26  BOTH ERRORS 2 AND 6 ABOVE OCCURRED.
```

```
       SUBROUTINE SIGTAB(NG,LG,MG,AG,BG,CG,DG,DGNUL,IGOPTA,FIRSTG,SAVEG,
      1                  EVRG,EVIG,IGERRA,RCOND,
      2                  OMEGA,RCONDS,ISOPTA,G,SIGMA,INFO,
      3                  RDUM,CDUM,IDUM,JERR,
      4                  JOP,NO,OMGTAB,SVMAX,SVMIN,KERR)
       COMPLEX
      + CDUM(*), G(MG,*)
       INTEGER
      + JERR, JOP, KERR, LG, MG, NG, NO, IDUM(*), IGERRA(2), IGOPTA(2),
      + INFO, ISOPTA(3)
       LOGICAL
      + DGNUL, FIRSTG
       REAL
      + OMEGA, RCOND, RCONDS, AG(*), BG(*), CG(*), DG(*), EVIG(*),
      + EVRG(*), OMGTAB(*), RDUM(*), SAVEG(*), SIGMA(*), SVMAX(*),
      + SVMIN(*)
C
C      THE PURPOSE OF  SIGTAB  IS TO REPEATEDLY CALL  SIGCAL
C      IN ITS  "GO = CG*(SQRT(-1)*OMEGA*I - AG)INVERSE*BG + DG"  MODE
C      WHILE VARYING THE VALUE OF THE FREQUENCY PARAMETER
C      "OMEGA" SO AS TO TABULATE THE MAXIMUM AND MINIMUM
C      SINGULAR VALUES OF THE "G" MATRIX WHICH  SIGCAL  CHOOSES TO
C      CALCULATE BASED ON THE VALUE OF THE PARAMETER "ISOPTA(1)".
C      THESE COMMENTS ASSUME THAT THE READER IS FAMILIAR WITH
C      THE PROGRAM PRELUDE DOCUMENTATION OF  SIGCAL.
C
C      TWO OF THE 27 PARAMETERS TO  SIGCAL  ARE COMPLEX MATRICES,
C      IDENTIFIED IN  SIGCAL  DOCUMENTATION AS "U" AND "V".  THESE
C      ARE USED WHEN  SIGCAL  IS CALLED WITH "ISOPTA(2)" SET EQUAL TO 3
C      TO RETURN LEFT AND RIGHT SINGULAR VECTORS OF "G".  SINCE
C      SIGTAB  CALLS  SIGCAL  WITH "ISOPTA(2)" SET TO 2,  SIGTAB
C      DOES NOT NEED "U" AND "V".  THE FIRST 25 PARAMETERS OF
C      SIGTAB, I.E., "NG" THROUGH "JERR" ARE IDENTICAL IN ORDER
C      AND FUNCTION (AND IN NAME IN THIS DOCUMENT) WITH THE
C      REMAINING 25 PARAMETERS TO  SIGCAL  WITH THE EXCEPTION THAT ONLY
C      THE FIRST TWO ENTRIES OF  IGOPTA  ARE FUNCTIONAL; EVEN IF THE
C      THIRD ENTRY IS PRESENT, IT IS IGNORED AND  SIGCAL  IS DRIVEN
C      IN ITS "COMPUTE GO INTERNALLY" MODE.
C
C      SIGTAB  SETS THE VALUE OF ISOPTA(3) TO 0 AFTER THE FIRST
C      CALL TO  SIGCAL  SO THAT DETAIL PRINTOUT OCCURS ONLY
C      WHEN  SIGCAL  IS CALLED WITH OMEGA = OMGTAB(1), AND
C      THAT ONLY IF THE USER SET  ISOPTA(3) .NE. 0  ON ENTRY
C      TO  SIGTAB.
C
C      FIRST 25 PARAMETERS
C          ON ENTRY, THE USER MUST INITIALIZE PARAMETERS NG, RG, MG, AG,
C              BG, CG, DG, DGNUL, IGOPTA(1), IGOPTA(2), FIRSTG,
C            ISOPTA(1), AND ISOPTA(3) IN ACCORDANCE WITH  SIGCAL
C            DOCUMENTATION.
C        ON RETURN,
C            IF KERR = -1, UNCHANGED
C            IF KERR = 0, AS RETURNED BY  SIGCAL  WITH
C                  OMEGA = OMGTAB(NO) EXCEPT ISOPTA(2) HAS BEEN SET TO
C                  2 AND ISOPTA(3) HAS BEEN SET TO 0
C            IF KERR > 0, AS RETURNED BY  SIGCAL  WITH
C                  OMEGA = OMGTAB(KERR) EXCEPT ISOPTA(2) HAS BEEN SET
C                  TO 2 AND ISOPTA(3) HAS BEEN SET TO 0
C
C    JOP     INTEGER, OPTION ON INPUT OMEGAS
C        ON ENTRY,
C            JOP = 0,  USER HAS PRESET THE ENTIRE OMGTAB ARRAY
C            JOP = 1,  USER HAS PRESET OMGTAB(1) AND OMGTAB(NO).
C                  SIGTAB  WILL FILL IN INTERMEDIATE VALUES
C                  EQUALLY SPACED
C            JOP = 2,  USER HAS PRESET OMGTAB(1) AND OMGTAB(NO)
C                  TO NONZERO VALUES OF THE SAME SIGN.  SIGTAB
C                  WILL FILL IN INTERMEDIATE VALUES SO THAT CONSECUTIVE
C                  RATIOS ARE THE SAME (EQUALLY SPACED ON A LOG SCALE).
C        ON RETURN, JOP = 0
C
C    NO      INTEGER
C        ON ENTRY, NUMBER OF ENTRIES IN THE OMGTAB, SVMAX, AND
C            SVMIN ARRAYS.
C        ON RETURN, UNCHANGED
C
C    OMGTAB  REAL ARRAY OF LENGTH AT LEAST  NO
C        ON ENTRY, PRESET AS DESCRIBED UNDER JOP
C        ON RETURN, FULLY SET ACCORDING TO JOP OPTION
C
C    SVMAX, SVMIN  REAL ARRAYS OF LENGTH AT LEAST  NO
C        ON ENTRY, NOT USED
C        ON RETURN, SVMAX(I) AND SVMIN(I) ARE SET TO, RESPECTIVELY,
C            THE MAXIMUM AND MINIMUM SINGULAR VALUES OF THE
C            G MATRIX CALCULATED BY  SIGCAL  WHEN OMEGA IS
C            SET TO OMGTAB(I).
C
C    KERR    INTEGER, ERROR PARAMETER OF SIGTAB
C        ON ENTRY, NOT USED
C        ON RETURN,
C            KERR = 0,  NO ERROR DETECTED
C            KERR = -1,  ERROR DETECTED IN INPUT VALUES OF
C                  NO, JOP, OR OMGTAB
C            KERR > 0,  SIGCAL DETECTED AN ERROR WHILE CALCULATING
C                  WITH OMEGA = OMGTAB(KERR).  IN THIS CASE, THE
C                  FIRST 23 PARAMETERS HAVE THE VALUES RETURNED
```

```
C                  BY  SIGCAL  WHEN THE ERROR OCCURRED.  IF
C                  KERR .GT. 1, THE FIRST KERR - 1 ENTRIES OF
C                  SVMAX AND SVMIN WERE CALCULATED WITHOUT ANY
C                  ERROR BEING DETECTED.
```

```
      SUBROUTINE SDTAB(NG,LG,MG,AG,BG,CG,DG,DGNUL,IGOPTA,FIRSTG,SAVEG,
     1                 EVRG,EVIG,GO,IGERRA,RCONDG,
     2                 NK,LK,MK,AK,BK,CK,DK,DKNUL,IKOPTA,FIRSTK,SAVEK,
     3                 EVRK,EVIK,KO,IKERRA,RCONDK,
     4                 OMEGA,RCONDS,ISOPTA,G,SIGMA,INFO,
     5                 RDUM,CDUM,IDUM,JERR,
     6                 JOP,NO,OMGTAB,SVMAX,SVMIN,KERR)
      COMPLEX
     + CDUM(*), G(*), GO(MG,*), KO(MK,*)
      INTEGER
     + JERR, JOP, KERR, MG, MK, NG, NK, NO, LG, LK, IDUM(*), IGERRA(2),
     + IGOPTA(2), IKERRA(2), IKOPTA(2), INFO, ISOPTA(4)
      LOGICAL
     + DGNUL, DKNUL, FIRSTG, FIRSTK
      REAL
     + OMEGA, RCONDG, RCONDK, RCONDS, AG(*), AK(*), BG(*), BK(*), CG(*),
     + CK(*), DG(*), DK(*), EVIG(*), EVIK(*), EVRG(*), EVRK(*),
     + OMGTAB(*), RDUM(*), SAVEG(*), SAVEK(*), SIGMA(*), SVMAX(*),
C    + SVMIN(*)
C
C     THE PURPOSE OF  SDTAB  IS TO REPEATEDLY CALL  SIGDYN
C     IN ITS  "CALCULATE BOTH  GO  AND  KO  INTERNALLY"  MODE
C     WHILE VARYING THE VALUE OF THE FREQUENCY PARAMETER
C     "OMEGA" SO AS TO TABULATE THE MAXIMUM AND MINIMUM
C     SINGULAR VALUES OF THE "G" MATRIX WHICH  SIGDYN  CHOOSES TO
C     CALCULATE BASED ON THE VALUE OF THE PARAMETER "ISOPTA(1)".
C     THESE COMMENTS ASSUME THAT THE READER IS FAMILIAR WITH
C     THE PROGRAM PRELUDE DOCUMENTATION OF  SIGDYN.
C
C
C     COMMENTS ARE INTERSPERSED THROUGHOUT THE CODE TO AID THE USER WHO
C     WISHES TO WRITE A ROUTINE TO TABULATE SINGULAR VALUES FOR A SYSTEM
C     WITH ONE OR BOTH OF THE  GO  AND  KO  MATRICES CALCULATED
C     EXTERNALLY. SEE APPENDIX A (P. 22).
C
C     TWO OF THE 44 PARAMETERS TO  SIGDYN  ARE COMPLEX MATRICES,
C     IDENTIFIED IN  SIGDYN  DOCUMENTATION AS "U" AND "V".  THESE
C     ARE USED WHEN  SIGDYN  IS CALLED WITH "ISOPTA(2)" SET EQUAL TO 3
C     TO RETURN LEFT AND RIGHT SINGULAR VECTORS OF "G".  SINCE
C     SDTAB  CALLS  SIGDYN  WITH "ISOPTA(2)" SET TO 2,  SDTAB
C     DOES NOT NEED "U" AND "V".  THE FIRST 42 PARAMETERS OF
C     SDTAB, I.E., "NG" THROUGH "JERR", ARE IDENTICAL IN ORDER
C     AND FUNCTION (AND IN NAME IN THIS DOCUMENT) WITH THE
C     REMAINING 42 PARAMETERS TO  SIGDYN  WITH THE EXCEPTION THAT ONLY
C     THE FIRST TWO ENTRIES OF  IGOPTA  AND  IKOPTA  ARE FUNCTIONAL;
C     EVEN IF THE THIRD ENTRIES ARE PRESENT THEY ARE IGNORED AND  SIGCAL
C     IS DRIVEN IN ITS  "COMPUTE  GO  AND  KO  INTERNALLY"  MODE.
C
C     SDTAB  SETS THE VALUE OF ISOPTA(3) TO 0 AFTER THE FIRST
C     CALL TO  SIGDYN  SO THAT DETAIL PRINTOUT OCCURS ONLY
C     WHEN  SIGDYN  IS CALLED WITH OMEGA = OMGTAB(1), AND
C     THAT ONLY IF THE USER SET  ISOPTA(3) .NE. 0  ON ENTRY
C     TO  SDTAB.
C
```

```
C    FIRST 42 PARAMETERS
C        ON ENTRY, THE USER MUST INITIALIZE PARAMETERS NG, LG, MG, AG,
C            BG, CG, DG, DGNUL, IGOPTA(1), IGOPTA(2), FIRSTG, NK, LK,
C            MK, AK, BK, CK, DK, DKNULL, IKOPTA(1), IKOPTA(2), FIRSTK,
C            ISOPTA(1), ISOPTA(3), AND ISOPTA(4) IN ACCORDANCE
C            WITH  SIGDYN  DOCUMENTATION.
C        ON RETURN,
C            IF KERR = -1, UNCHANGED
C            IF KERR = 0, AS RETURNED BY  SIGDYN  WITH
C                  OMEGA = OMGTAB(NO) EXCEPT ISOPTA(2) HAS BEEN SET TO
C                  2 AND ISOPTA(3) HAS BEEN SET TO 0
C            IF KERR > 0, AS RETURNED BY  SIGDYN  WITH
C                  OMEGA = OMGTAB(KERR) EXCEPT ISOPTA(2) HAS BEEN SET
C                  TO 2 AND ISOPTA(3) HAS BEEN SET TO 0
C
C    JOP      INTEGER, OPTION ON INPUT OMEGAS
C        ON ENTRY,
C            JOP = 0,  USER HAS PRESET THE ENTIRE OMGTAB ARRAY
C            JOP = 1,  USER HAS PRESET OMGTAB(1) AND OMGTAB(NO).
C                  SDTAB  WILL FILL IN INTERMEDIATE VALUES
C                  EQUALLY SPACED
C            JOP = 2,  USER HAS PRESET OMGTAB(1) AND OMGTAB(NO)
C                  TO NONZERO VALUES OF THE SAME SIGN.  SDTAB
C                  WILL FILL IN INTERMEDIATE VALUES SO THAT CONSECUTIVE
C                  RATIOS ARE THE SAME (EQUALLY SPACED ON A LOG SCALE).
C        ON RETURN, JOP = 0
C
C    NO       INTEGER
C        ON ENTRY, NUMBER OF ENTRIES IN THE OMGTAB, SVMAX, AND
C            SVMIN ARRAYS.
C        ON RETURN, UNCHANGED
C
C    OMGTAB  REAL ARRAY OF LENGTH AT LEAST  NO
C        ON ENTRY, PRESET AS DESCRIBED UNDER JOP
C        ON RETURN, FULLY SET ACCORDING TO JOP OPTION
C
C    SVMAX, SVMIN  REAL ARRAYS OF LENGTH AT LEAST  NO
C        ON ENTRY, NOT USED
C        ON RETURN, SVMAX(I) AND SVMIN(I) ARE SET TO, RESPECTIVELY,
C            THE MAXIMUM AND MINIMUM SINGULAR VALUES OF THE
C            G MATRIX CALCULATED BY  SIGDYN  WHEN OMEGA IS
C            SET TO OMGTAB(I).
C
C    KERR     INTEGER, ERROR PARAMETER OF SDTAB
C        ON ENTRY, NOT USED
C        ON RETURN,
C            KERR = 0,  NO ERROR DETECTED
C            KERR = -1,  ERROR DETECTED IN INPUT VALUES OF
```

```
C                     NO, JOP, OR OMGTAB
C              KERR > 0,  SIGDYN DETECTED AN ERROR WHILE CALCULATING
C                     WITH OMEGA = OMGTAB(KERR).  IN THIS CASE, THE
C                     FIRST 38 PARAMETERS HAVE THE VALUES RETURNED
C                     BY  SIGDYN  WHEN THE ERROR OCCURRED.  IF
C                     KERR .GT. 1, THE FIRST KERR - 1 ENTRIES OF
C                     SVMAX AND SVMIN WERE CALCULATED WITHOUT ANY
C                     ERROR BEING DETECTED.
```

```
      SUBROUTINE TABGEN(NO,OMGTAB,JOP,KERR)
       INTEGER
     + JOP, KERR, NO
       REAL
     + OMGTAB(NO)
C
C     SUBROUTINE  TABGEN  GENERATES A TABLE OF NUMBERS SPACED WITH THE
C     USER'S CHOICE OF EQUAL INCREMENTS OR EQUAL CONSECUTIVE RATIOS.
C
C     PARAMETERS
C
C     NO         INTEGER
C                ON ENTRY,  NUMBER OF LOCATIONS IN OMGTAB.
C                ON RETURN, UNCHANGED.
C
C     OMGTAB     REAL ARRAY OF AT LEAST  NO  LOCATIONS.
C                ON ENTRY,  OMGTAB(1) AND OMGTAB(NO) MUST BE SET TO THE
C                   FIRST AND LAST ENTRIES OF THE DESIRED TABLE.
C                ON NORMAL RETURN, THE REMAINDER OF OMGTAB IS FILLED IN
C                   ACCORDING TO THE OPTION REQUESTED BY PARAMETER  JOP.
C
C     JOP        INTEGER
C                ON ENTRY,
C                   IF JOP = 0, CHECK THAT  NO  IS POSITIVE,  DO NOT ALTER
C                     OMGTAB.
C                   IF JOP = 1, FILL IN OMGTAB SO THAT CONSECUTIVE
C                     DIFFERENCES ARE EQUAL.
C                   IF JOP = 2, FILL IN OMGTAB SO THAT CONSECUTIVE
C                     RATIOS ARE EQUAL.
C                ON NORMAL RETURN, JOP = 0.
C
C     KERR       INTEGER
C                ON ENTRY, NOT USED.
C                ON RETURN,
C                   KERR = 0, NORMAL RETURN, NO ERROR DETECTED.
C                   KERR = -1, ERROR RETURN, OMGTAB AND JOP UNCHANGED.
C                     POSSIBLE CAUSES:  NO  NONPOSITIVE; JOP NOT EQUAL TO
C                       0, 1, OR 2; JOP = 2 BUT OMGTAB(1) AND OMGTAB(NO)
C                       DO NOT HAVE THE SAME SIGN.
```

Appendix C

Symbolic Names of Global Entities

In this appendix the global names used in FREQ are listed. The user's application program that uses FREQ must not use any of these names for global entities (main program name, subroutine name, function name, common block name, etc.). We also tell where each of the FREQ global entities belongs in the software structure of figures 2 and 3.

The single-precision version of FREQ uses the global names:

BALANC	CHECO	CSCAL	CUNITY	MULT	SCL1	SIGCAL
CADD	CHEFA	CSOLVE	EQUATE	ORACCH	SCNRM2	SIGDYN
CAXPY	CHESL	CSROT	FRDTIT	ORACES	SDTAB	SIGTAB
CDOTC	CL1NRM	CSSCAL	FREQNT	ORACIO	SFRMG	SROTG
CEQUAT	CMMULT	CSUBT	FREQNV	ORACPC	SHETR	SUBT
CGECO	CMULT	CSVDC	HQR	ORCERP	SIGCA1	TABGEN
CGEFA	CPRNT	CSWAP	ICAMAX	PRNT	SIGCA2	
CGESL	CRADD	CTRANP	LNCNT	SCASUM	SIGCA3	

The double-precision version of FREQ uses the global names:

BALANC	CSUBT	DZASUM	LNCNT	SDTAB	TABGEN	ZHECO
CADD	CTRANP	DZNRM2	MULT	SIGCA1	ZAXPY	ZHEFA
CEQUAT	CUNITY	EQUATE	ORACCH	SIGCA2	ZDOTC	ZHESL
CMMULT	DCABS1	FRDTIT	ORACES	SIGCA3	ZDROT	ZL1NRM
CMULT	DCL1	FREQNT	ORACIO	SIGCAL	ZDSCAL	ZSCAL
CPRNT	DFRMG	FREQNV	ORACPC	SIGDYN	ZGECO	ZSVDC
CRADD	DHETR	HQR	ORCERP	SIGTAB	ZGEFA	ZSWAP
CSOLVE	DROTG	IZAMAX	PRNT	SUBT	ZGESL	

These global entities break down in the following manner. With reference to figure 2, the top three boxes show the initialization routines. They are FREQNT, FRDTIT, and FREQNV. The output routines are ORACES, LNCNT, ORCERP, PRNT, and CPRNT. The common blocks are /ORACIO/, /ORACPC/, and /ORACCH/.

Everything else falls in the category "Computational routines." With reference to figure 3, the top five boxes show the user interface routines. They are SIGCAL, SIGDYN, SIGTAB, SDTAB, and TABGEN. The subordinate routines are SIGCA1, SIGCA2, SIGCA3, CSOLVE, CRADD, CEQUAT, CTRANP, CUNITY, CADD, CSUBT, CMULT, CMMULT, EQUATE, SUBT, and MULT.

In the single-precision version of FREQ, Laub's frequency-response subroutines are CHECO, CHEFA, CHESL, CL1NRM, SCL1, SFRMG, and SHETR; whereas in the double-precision version, they are DFRMG, DHETR, ZHECO, ZHEFA, ZHESL, ZL1NRM, and DCL1.

The EISPACK subroutines are BALANC and HQR. In single precision, the LINPACK subroutines are CGECO, CGEFA, CGESL, CSVDC, CAXPY, CDOTC, CSCAL, CSROT, CSSCAL, CSWAP, ICAMAX, SCASUM, SCNRM2, and SROTG; whereas in double precision, they are ZGECO, ZGEFA, ZGESL, ZSVDC, DCABS1, DROTG, DZASUM, DZNRM2, ZDROT, IZAMAX, ZAXPY, ZDOTC, ZDSCAL, ZSCAL, and ZSWAP.

The global names LNCNT and PRNT from the output group and EQUATE, SUBT, and MULT from the subordinate routines group also occur in the software package ORACLS. The FREQ versions have the same function and (except that the fourth parameter of PRNT is now of type CHARACTER) the same calling sequences as their ORACLS counterparts. They have been modified to be compatible with FREQ output and error-processing protocols.

Appendix D

Example Program Listing

This appendix presents a listing of the program that gives rise to the example presented previously.

```
      PROGRAM HOOPCA
C
C**** SPECIFY VARIABLES AND ARRAYS
C
C     PARAMETER TO SIZE THE FREQUENCY TABLE
C
      INTEGER
     + NOMAX
      PARAMETER (NOMAX=400)
C
C     DATA ARRAYS TO BUILD PLANT AND COMPENSATOR MATRICES AND
C     VARIABLES AND ARRAYS NEEDED FOR FREQ
C
      CHARACTER*80
     + TITLE
      COMPLEX
     + CDUM(780), GC(3,3), GO(3,3), KO(3,3)
      INTEGER
     + I, IDUM(26), IEROPT, IGERRA(2), IGOPTA(2), IKERRA(2), IKOPTA(2),
     + INFO, ISOPTA(4), IUL, J, JERR, JOP, JUL, KERR, LD, LG, LK, LP,
     + MD, MG, MK, MP, NCL, ND, ND1(2), NERR, NG, NIN, NK, NLP, NO,
     + NOUT, NP, NPLOT, NVC(2), N102(2), N1212(2), N123(2), N1414(2),
     + N143(2), N2020(2), N203(2), N22(2), N2626(2), N263(2), N312(2),
     + N314(2), N326(2), N33(2)
      LOGICAL
     + DDNUL, DGNUL, DKNUL, DPNUL, FIRSTD, FIRSTG, FIRSTK, FIRSTP
      REAL
     + AD(14,14), AE(2,2), AG(26,26), AK(12,12), AP(12,12), BD(14,3),
     + BG(26,3), BK(12,3), BP(12,3), CD(3,14), CG(3,26), CK(3,12),
     + CP(3,12), DD, DG(1), DK(1), DP, D1(144), EVIG(26), EVIK(12),
     + EVRG(26), EVRK(12), E3(3,3), FNDR(10,2), G(3,12), GT(12,3),
     + H(12,3), OMEGA, OMGTAB(NOMAX), RCONDG, RCONDK, RCONDS, RDUM(26),
     + RINV(3,3), SAVED(280), SAVEG(832), SAVEK(216), SAVEP(216),
     + SIGMA(26), SVMAX(NOMAX,11), SVMIN(NOMAX,11), U(20,20), UI(2,2),
     + VB(20,3), VC(3,20)
C
C**** INITIALIZATION OF FILES AND COMMON BLOCKS IN FREQ
C
C     OPEN FILE 'HCAEGD' FOR USE AS FREQ INPUT FILE AND FILE 'HCAEGO'
```

```
C    FOR USE AS FREQ OUTPUT AND ERROR MESSAGE FILE
C
     OPEN(5,FILE='HCAEGD',STATUS='OLD')
     OPEN(6,FILE='HCAEGO',STATUS='UNKNOWN')
     REWIND 5
     REWIND 6
C
C    SET FREQ PARAMETERS TO DEFAULT VALUES
C
     CALL FREQNT
C
C    ENTER THOSE DEFAULT VALUES INTO USER VARIABLES
C
     CALL FREQNV(-1,NIN,NOUT,NERR,NLP,NCL,IEROPT,TITLE)
C
C    IN THE USER VARIABLES, REPLACE THE DEFAULT VALUES BY USER-
C    SELECTED VALUES
     NLP = 55
     NCL = 80
     TITLE = 'DEMONSTRATION OF FREQ, BASED ON EXAMPLE IN '
    1 //'NASA TP-2560'
C
C    INSTRUCT FREQ TO USE THE NEW VALUES
C
     CALL FREQNV(+1,NIN,NOUT,NERR,NLP,NCL,IEROPT,TITLE)
C
C**** READ IN DATA MATRICES
C
C    THE CALL TO LNCNT TELLS THE FREQ OUTPUT LINE COUNTER HOW MANY
C    LINES THE USER IS PUTTING ON THE FREQ OUTPUT FILE
C
     CALL LNCNT(12)
     WRITE (6,100)
 100 FORMAT(/1X,'DATA FROM NASA TP-2560 USED TO GENERATE LINEAR SYSTEM'
    + ,' MATRICES FOR',/,
    + 6X,'THE 26TH-ORDER PLANT',/,
    + 6X,'THE 12TH-ORDER DESIGN MODEL',/,
    + 6X,'THE 14TH-ORDER ERROR (DELTA G) SYSTEM',/,
    + 6X,'THE 12TH-ORDER ATTITUDE FEEDBACK COMPENSATOR',//,
    + 6X,'"FNDR" CONTAINS THE FREQUENCIES (HZ) AND THE DAMPING RATIOS'
    + ,/,6X,
    + '"VB" CONTAINS MODE SLOPE INFORMATION FOR "B" AND "C" MATRICES'
    + ,/,6X,'"H" IS THE KALMAN FILTER GAINS MATRIX',/,
    + 6X,'"GT" IS THE TRANSPOSE OF THE FEEDBACK GAINS MATRIX',/)
     CALL READ(5,FNDR,N102,RINV,N33,VB,N203,H,N123,GT,N123)
C
C**** BUILD PLANT MATRICES AS IN THE APPENDIX OF NASA TP-2560
C
```

```
C     E3 IS THE 3 BY 3 IDENTITY MATRIX
C
      CALL UNITY(E3,N33)
C
C     INITIALIZE AG
C
      N2626(1) = 26
      N2626(2) = 26
      CALL NULL(AG,N2626)
C
C     INSERT IDENTITY IN AG (LOC. CIT., EQUATIONS (A3) AND (A4))
C
      CALL MERGE(E3,N33,AG,N2626,1,4)
C
C     INSERT FLEXIBLE MODES IN AG
C         (LOC. CIT., EQUATIONS (A3), (A5), (A6))
C
      AE(1,1) = 0.0E+00
      AE(1,2) = 1.0E+00
      N22(1) = 2
      N22(2) = 2
      DO 10 I = 1,10
      AE(2,1) = -FNDR(I,1)**2
      AE(2,2) = -2.0E+00*FNDR(I,1)*FNDR(I,2)
      IUL = 2*I + 5
      JUL = IUL
      CALL MERGE (AE,N22,AG,N2626,IUL,JUL)
   10 CONTINUE
C
C     INITIALIZE BG
C
      N263(1) = 26
      N263(2) = 3
      CALL NULL(BG,N263)
C
C     INSERT INVERSE OF INERTIA MATRIX (LOC. CIT., EQN (A7))
C
      CALL MERGE(RINV,N33,BG,N263,4,1)
C
C      INSERT ACTUATOR EFFECTS ON FLEXIBLE MODES (IBID)
C
      CALL MERGE(VB,N203,BG,N263,7,1)
C
C      INITIALIZE CG
C
      N326(1) = 3
      N326(2) = 26
      CALL NULL(CG,N326)
```

```
C
C     INSERT IDENTITY IN CG (LOC. CIT., EQN (A11))
C
      CALL MERGE(E3,N33,CG,N326,1,1)
C
C     FORM VC (LOC. CIT., EQNS (A12), (A13), (A14))
C
      UI(1,1) = 0.0E+00
      UI(1,2) = 1.0E+00
      UI(2,1) = 1.0E+00
      UI(2,2) = 0.0E+00
      N2020(1) = 20
      N2020(2) = 20
      CALL NULL(U,N2020)
      DO 20 I = 1,10
      IUL = 2*I-1
      JUL = IUL
      CALL MERGE(UI,N22,U,N2020,IUL,JUL)
   20 CONTINUE
      CALL MULT(U,N2020,VB,N203,D1,ND1)
      CALL TRANP(D1,ND1,VC,NVC)
      CALL MERGE(VC,NVC,CG,N326,1,7)
C
C**** BUILD COMPENSATOR MATRICES AS IN NASA TP-2560, SECTIONS 5 AND 6
C      DESIGN MODEL IS THE FIRST 12 STATES OF THE PLANT
C      COMPENSATOR USES ATTITUDE FEEDBACK
C      COMPENSATOR GAINS FROM LOC. CIT., TABLE II
C
C      GENERATE AP, BP, AND CP MATRICES
C
       N1212(1) = 12
       N1212(2) = 12
       N312(1) = 3
       N312(2) = 12
       CALL EXTRCT(AG,N2626,AP,N1212,1,1)
       CALL EXTRCT(BG,N263,BP,N123,1,1)
       CALL EXTRCT(CG,N326,CP,N312,1,1)
C
C      GENERATE G FROM THE G TRANSPOSED WHICH WAS INPUT
C
       CALL TRANP(GT,N123,G,N312)
C
C      BUILD AK AS AP-BP*G-H*CP (FROM LOC. CIT., P5, LAST EQUATION)
C
       CALL MULT(BP,N123,G,N312,D1,ND1)
       CALL SUBT(AP,N1212,D1,ND1,AK,ND1)
       CALL MULT(H,N123,CP,N312,D1,ND1)
       CALL SUBT(AK,N1212,D1,ND1,AK,N1212)
```

```
C
C     BK AND CK ARE H AND G, RESPECTIVELY (IBID.)
C
      CALL EQUATE(H,N123,BK,N123)
      CALL EQUATE(G,N312,CK,N312)
C
C     DETERMINE THE MATRICES FOR THE DELTA G OF LOC. CIT., FIGURE 17
C
      N1414(1) = 14
      N1414(2) = 14
      N143(1) = 14
      N143(2) = 3
      N314(1) = 3
      N314(2) = 14
      CALL EXTRCT(AG,N2626,AD,N1414,13,13)
      CALL EXTRCT(BG,N263,BD,N143,13,1)
      CALL EXTRCT(CG,N326,CD,N314,1,13)
C
C**** FIRST WE WILL DO THE OPEN-LOOP SINGULAR VALUE CALCULATIONS
C     FOR SEVERAL OF THESE SYSTEMS.  HERE WE SET PARAMETERS WHICH
C     WILL BE THE SAME FOR SEVERAL CALLS TO SIGTAB.
C
C     BALANCE SYSTEM MATRIX AND CALCULATE EIGENVALUES
C
     IGOPTA(1) = 1
C
C     ESTIMATE RECIPROCAL CONDITION NUMBER WHEN CALCULATING LOOP-GAIN
C     TRANSFER MATRIX
C
     IGOPTA(2) = -1
C
C     CALCULATE LOOP-GAIN TRANSFER MATRIX
C
      ISOPTA(1) = 1
C
C     CALCULATE SINGULAR VALUES AND NO SINGULAR VECTORS
C         (SINCE WE WILL CALL SIGTAB, THIS CHOICE WOULD BE FORCED
C         ON US ANYWAY)
C
      ISOPTA(2) = 2
C
C     USE A PROBLEM-DEPENDENT SUBROUTINE TO SET NO AND OMGTAB.
C     IN THIS CASE, WE MERGE THE GRID USED FOR THE PLOTS IN TP-2560 WITH
C     301 POINTS DISTRUBUTED OVER THE 3 DECADES WE ARE USING (FROM .01 HZ
C     TO 10 HZ) SO THAT RATIOS OF CONSECUTIVE GRID VALUES ARE ALL THE
C     SAME.  THESE POINTS WILL BE EQUALLY SPACED ALONG THE LOG AXIS USED
C     FOR PLOTTING SINGULAR VALUES AS A FUNCTION OF FREQUENCY.
C
```

```
      CALL OMGEN(NOMAX,NO,OMGTAB)
C
C     TELL SIGTAB THAT THE OMGTAB ARRAY IS PRESET
C
      JOP = 0
C
C**** NOW SET THE PARAMETERS NEEDED BY SIGTAB TO CALCULATE THE OPEN-
C     LOOP FREQUENCY RESPONSE OF THE PLANT
C
      NG = 26
      LG = 3
      MG = 3
      DGNUL = .TRUE.
      FIRSTG = .TRUE.
      ISOPTA(3) = 1
      CALL LNCNT(4)
      WRITE (6,130)
  130 FORMAT(//,1X,'NEXT CALCULATE THE OPEN-LOOP RESPONSE OF THE ',
     + 'FULL 26TH-ORDER PLANT',/)
      CALL SIGTAB(NG,LG,MG,AG,BG,CG,DG,DGNUL,IGOPTA,FIRSTG,SAVEG,
     1            EVRG,EVIG,IGERRA,RCONDG,
     2            OMEGA,RCONDS,ISOPTA,GC,SIGMA,INFO,
     3            RDUM,CDUM,IDUM,JERR,
     4            JOP,NO,OMGTAB,SVMAX(1,1),SVMIN(1,1),KERR)
C
C**** NOW SET THE PARAMETERS NEEDED BY SIGTAB TO CALCULATE THE OPEN-
C     LOOP FREQUENCY RESPONSE OF THE DESIGN MODEL
C
      NP = 12
      LP = 3
      MP = 3
      DPNUL = .TRUE.
      FIRSTP = .TRUE.
      ISOPTA(3) = 1
      CALL LNCNT(4)
      WRITE (6,110)
  110 FORMAT(//,1X,'NEXT CALCULATE THE OPEN-LOOP RESPONSE OF THE ',
     + '12TH-ORDER DESIGN MODEL',/)
      CALL SIGTAB(NP,LP,MP,AP,BP,CP,DP,DPNUL,IGOPTA,FIRSTP,SAVEP,
     1            EVRG,EVIG,IGERRA,RCONDG,
     2            OMEGA,RCONDS,ISOPTA,GC,SIGMA,INFO,
     3            RDUM,CDUM,IDUM,JERR,
     4            JOP,NO,OMGTAB,SVMAX(1,2),SVMIN(1,2),KERR)
C
C**** NOW SET THE PARAMETERS NEEDED BY SIGTAB TO CALCULATE THE OPEN-
C     LOOP FREQUENCY RESPONSE OF THE "DELTA G" (ADDITIVE ERROR) SYSTEM
C
C     LINEAR SYSTEM SIZING PARAMETERS
```

```
      ND = 14
      LD = 3
      MD = 3
      DDNUL = .TRUE.
      FIRSTD = .TRUE.
      ISOPTA(3) = 1
      CALL LNCNT(4)
      WRITE (6,120)
  120 FORMAT(//,1X,'NEXT CALCULATE THE SINGULAR VALUE PLOTS OF THE',
     + ' 14TH-ORDER ERROR SYSTEM',/)
      CALL SIGTAB(ND,LD,MD,AD,BD,CD,DD,DDNUL,IGOPTA,FIRSTD,SAVED,
     1            EVRG,EVIG,IGERRA,RCONDG,
     2            OMEGA,RCONDS,ISOPTA,GC,SIGMA,INFO,
     3            RDUM,CDUM,IDUM,JERR,
     4            JOP,NO,OMGTAB,SVMAX(1,3),SVMIN(1,3),KERR)
C
C**** NOW SET THE PARAMETERS NEEDED BY SIGTAB TO CALCULATE THE OPEN-
C     LOOP FREQUENCY RESPONSE OF THE COMPENSATOR
C
      NK = 12
      LK = 3
      MK = 3
      DKNUL = .TRUE.
      IKOPTA(1) = 1
      IKOPTA(2) = -1
      FIRSTK = .TRUE.
      ISOPTA(3) = 1
      CALL LNCNT(4)
      WRITE (6,140)
  140 FORMAT(//,1X,'NEXT CALCULATE THE OPEN-LOOP RESPONSE OF THE ',
     + 'COMPENSATOR',/)
      CALL SIGTAB(NK,LK,MK,AK,BK,CK,DK,DKNUL,IKOPTA,FIRSTK,SAVEK,
     1            EVRK,EVIK,IKERRA,RCONDK,
     2            OMEGA,RCONDS,ISOPTA,GC,SIGMA,INFO,
     3            RDUM,CDUM,IDUM,JERR,
     4            JOP,NO,OMGTAB,SVMAX(1,4),SVMIN(1,4),KERR)
C
C**** SET THE PARAMETERS (NOT ALREADY SET) NEEDED BY SDTAB TO ANALYZE
C     THE DESIGN MODEL WITH COMPENSATOR.
C
C     TELL FREQ TO USE GO*KO FOR LOOP TRANSFER MATRIX
C
      ISOPTA(4) = 0
C     TURN ON PRINT (WAS TURNED OFF BY PREVIOUS CALL TO SIGTAB OR SDTAB)
      ISOPTA(3) = 1
C     TABULATE THE MAXIMUM AND MINIMUM SINGULAR VALUES OF THE OPEN- AND
C     CLOSED-LOOP DESIGN MODEL WITH COMPENSATOR.  NOTE THAT EACH CASE
C     HAS ITS OWN COLUMN IN OUTPUT ARRAYS.  ONLY THE OPEN-LOOP
```

```
C     CALCULATION WILL GENERATE PRINTOUT.
      DO 30 I = 1,2
C
C     FIRST TIME THROUGH THE DO LOOP, SINGULAR VALUE ANALYSIS WILL BE
C     APPLIED TO THE LOOP-GAIN TRANSFER MATRIX; SECOND TIME, THE
C     FEEDBACK TRANSFER MATRIX WILL BE USED.
C
      ISOPTA(1) = I
      IF (I .EQ. 1) CALL LNCNT(5)
      IF (I .EQ. 1) WRITE (6,150)
  150 FORMAT(//,1X,'NEXT CALCULATE THE OPEN-LOOP RESPONSE OF THE ',/,
     + 3X,'CASCADE OF COMPENSATOR AND 12TH-ORDER DESIGN MODEL',/)
      CALL SDTAB(NP,LP,MP,AP,BP,CP,DP,DPNUL,IGOPTA,FIRSTP,SAVEP,
     1           EVRG,EVIG,GO,IGERRA,RCONDG,
     2           NK,LK,MK,AK,BK,CK,DK,DKNUL,IKOPTA,FIRSTK,SAVEK,
     3           EVRK,EVIK,KO,IKERRA,RCONDK,
     4           OMEGA,RCONDS,ISOPTA,GC,SIGMA,INFO,
     5           RDUM,CDUM,IDUM,JERR,
     6           JOP,NO,OMGTAB,SVMAX(1,4+I),SVMIN(1,4+I),KERR)
   30 CONTINUE
C**** SET THE PARAMETERS (NOT ALREADY SET) NEEDED BY SDTAB TO ANALYZE
C     THE PLANT WITH COMPENSATOR.  BY EXECUTING A LOOP WHICH ALTERS THE
C     VALUE OF ISOPTA(1), ANALYSIS WILL BE PERFORMED USING THE
C         (1)  LOOP-GAIN TRANSFER MATRIX
C         (2)  FEEDBACK TRANSFER MATRIX
C         (3)  SENSITIVITY MATRIX
C         (4)  INVERSE RETURN DIFFERENCE MATRIX
C         (5)  RETURN DIFFERENCE MATRIX
C     OF THE TRANSFER FUNCTION GO*KO WHERE GO AND KO ARE, RESPECTIVELY,
C     THE PLANT AND COMPENSATOR TRANSFER MATRICES.  PARAMETERS WHICH
C     CAN BE ALTERED BY CALLING SDTAB WILL BE SET INSIDE THE LOOP,
C     OTHERS BEFORE ENTERING THE LOOP.
C
C     START LOOP
C
C     THE NEXT LINE FORCES THE SUBSEQUENT WRITE TO THE FREQ OUTPUT
C     UNIT TO START AT THE TOP OF A NEW PAGE
C
      CALL LNCNT(-1)
      CALL LNCNT(6)
C
C     THE FOLLOWING PRINTOUT INFORMS THE USER ABOUT THE LACK OF DETAILED
C     OUTPUT FOR THE SECOND TRIP THROUGH THE PREVIOUS "DO 30" LOOP AND
C     THE 5 TRIPS THROUGH THE FOLLOWING "DO 40" LOOP
C
      WRITE (6,160)
  160 FORMAT(//,1X,
     + 'PRINTOUT HAS BEEN TURNED OFF FOR THE NEXT 6 ANALYSES.',/,
     + 6X,'THESE ARE THE FEEDBACK TRANSFER MATRIX FOR THE DESIGN ',
     + 'MODEL WITH',/,6X,'COMPENSATOR; ',
     + 'AND ALL 5 ANALYSES OFFERED BY "FREQ" USING THE FULL',/,
     + 6X,'26TH-ORDER PLANT WITH THE COMPENSATOR')
      DO 40 I = 1,5
C
C     SELECT MATRIX TO ANALYZE
C
      ISOPTA(1) = I
C
C     TABULATE THE MAXIMUM AND MINIMUM SINGULAR VALUES OF THE SELECTED
C     MATRIX AS A FUNCTION OF FREQUENCY.  NOTE THAT EACH CASE HAS ITS
C     OWN COLUMN IN OUTPUT ARRAYS.
C
      CALL SDTAB(NG,LG,MG,AG,BG,CG,DG,DGNUL,IGOPTA,FIRSTG,SAVEG,
     1           EVRG,EVIG,GO,IGERRA,RCONDG,
     2           NK,LK,MK,AK,BK,CK,DK,DKNUL,IKOPTA,FIRSTK,SAVEK,
     3           EVRK,EVIK,KO,IKERRA,RCONDK,
     4           OMEGA,RCONDS,ISOPTA,GC,SIGMA,INFO,
     5           RDUM,CDUM,IDUM,JERR,
     6           JOP,NO,OMGTAB,SVMAX(1,I+6),SVMIN(1,I+6),KERR)
   40 CONTINUE
C
C**** WRITE OUT A DATA FILE CONTAINING THE SINGULAR VALUE INFORMATION
C
      OPEN(10,FILE='HCAEGS',STATUS='UNKNOWN')
      REWIND 10
      NPLOT = 11
      WRITE (10,'(2I20)') NO,NPLOT
      DO 50 I = 1,NO
      WRITE (10,'(4E20.14)') OMGTAB(I),(SVMAX(I,J),SVMIN(I,J),J=1,NPLOT)
   50 CONTINUE
      END
      SUBROUTINE OMGEN(NOMAX,NO,OMGTAB)
      INTEGER
     + NO, NOMAX
      REAL
     + OMGTAB(*)
      INTEGER
     + NOL, NXTRA
      PARAMETER (NOL = 70,NXTRA=301)
C
C**** PROBLEM-DEPENDENT SUBROUTINE TO GENERATE THE ARRAY OF FREQUENCY
C     VALUES AT WHICH THE ANALYSES ARE TO BE PERFORMED.
C     THIS IS A MERGING OF THE GRID USED FOR THE PLOTS IN TP-2560
C     WITH A 301-POINT GRID GEOMETRICALLY SPACED OVER THE SAME
C     3 DECADE INTERVAL FROM .01 HZ TO 10 HZ.
C
      CHARACTER*80
     + MSG
      INTEGER
     + I, JOP, KERR
      REAL
     + OM(NOL)
       DATA OM/.01,.03,.05,.07,.082,.09,.1,.15,.22,.27,.3,.4,.5,.6,.7,
     1 .75,.8,.9,1.,
     1 1.1,1.2,1.3,1.35,1.4,1.5,1.6,1.7,1.8,1.9,2.,2.5,3.,3.1,3.18,
     1  3.2,3.3,3.4,3.6,4.,4.3,4.5,4.7,5.,5.1,5.3,5.4,5.5,5.6,
     1  5.7,5.8,5.9,6.,6.4,6.6,6.7,6.8,6.9,7.,7.2,7.3,7.4,
     1  7.5,7.7,7.9,8.,8.6,8.78,8.9,9.,10./
C
C     IF THERE IS ENOUGH ROOM IN THE OMGTAB ARRAY
C
      IF (NOMAX .GE. NOL + NXTRA) THEN
C
C         SET NO AND ENTER THE FREQUENCY VALUES INTO OMGTAB
C
          NO = NOL + NXTRA
          DO 10 I = 1,NOL
            OMGTAB(I) = OM(I)
   10     CONTINUE
          OMGTAB(NOL+1) = OM(1)
          OMGTAB(NO) = OM(NOL)
          JOP = 2
          CALL TABGEN(NXTRA,OMGTAB(NOL+1),JOP,KERR)
          IF (KERR .NE. 0) THEN
            WRITE (MSG,40) KERR
   40       FORMAT('IN OMGEN, TABGEN KERR = ',I20)
            CALL ORCERP(3,MSG)
            STOP 'OMGEN FAILURE AT TABGEN'
          END IF
```

```
            CALL SRTNOD(NO,OMGTAB)
        ELSE
C
C           IF OMGTAB IS TOO SMALL, WRITE ERROR MESSAGE AND STOP
C
            WRITE (MSG,20) NOL
   20       FORMAT('OMGEN CANNOT PUT ',I10,' ENTRIES IN OMGTAB')
            CALL ORCERP(1,MSG)
            WRITE (MSG,30) NOMAX
   30       FORMAT('ONLY ',I10,' SPACES AVAILABLE')
            CALL ORCERP(2,MSG)
            STOP 'OMGEN FAILURE'
        END IF
        RETURN
        END
```

```
      SUBROUTINE SRTNOD(N,A)
      INTEGER
     + N
      REAL
     + A(*)
      INTEGER
     + I, J, K
      REAL
     + T
C
C**** SORT A(1),...,A(N) INTO INCREASING ORDER
C     IF THERE ARE DUPLICATES, ELIMINATE THEM AND REDUCE N ACCORDINGLY
C
      IF (N .LE. 1) RETURN
      K = 2
   10 CONTINUE
      IF (K .GT. N) RETURN
      T = A(K)
      DO 20 J = 1,K-1
      IF (T .LE. A(J)) GO TO 30
   20 CONTINUE
      K = K + 1
      GO TO 10
   30 CONTINUE
      IF (T .LT. A(J)) GO TO 40
      A(K) = A(N)
      N = N - 1
      GO TO 10
   40 CONTINUE
      DO 50 I = K-1,J,-1
      A(I+1) = A(I)
   50 CONTINUE
      A(J) = T
      GO TO 10
      END
      SUBROUTINE READ(I,A,NA,B,NB,C,NC,D,ND,E,NE)
      INTEGER
     + I, NA(2), NB(2), NC(2), ND(2), NE(2)
      REAL
     + A(*), B(*), C(*), D(*), E(*)
      COMMON /ORACIO/ NIN, NOUT, NERR
      INTEGER
     + NERR, NIN, NOUT
      SAVE /ORACIO/
      CHARACTER*4
     + LAB
      INTEGER
     + NZ(2)
      IF (I .LT. 1) GO TO 999
      READ(NIN,100,END=10000) LAB,            NZ(1), NZ(2)
      CALL READ1(A, NA,NZ,  LAB)
      IF(I .EQ. 1) GO TO 999
      READ(NIN,100,END=10000) LAB,            NZ(1), NZ(2)
      CALL READ1(B, NB,NZ,  LAB)
      IF(I .EQ. 2) GO TO 999
      READ(NIN,100,END=10000) LAB,            NZ(1), NZ(2)
      CALL READ1(C,NC,NZ,LAB)
      IF(I .EQ. 3) GO TO 999
      READ(NIN,100,END=10000) LAB,            NZ(1), NZ(2)
      CALL READ1(D, ND,NZ,  LAB)
      IF(I .EQ. 4) GO TO 999
      READ(NIN,100,END=10000) LAB,            NZ(1), NZ(2)
      CALL READ1(E, NE,NZ,  LAB)
  100 FORMAT(BZ,A4,4X,2I4)
  999 RETURN
10000 CALL ORCERP(3,'END OF FILE IN READ')
      END
      SUBROUTINE TRANP(A,NA,B,NB)
      INTEGER
     + NA(2), NB(2)
      REAL
     + A(*), B(*)
      CHARACTER*42
     + MSG
      INTEGER
     + I, IJ, IR, J, L, NC, NR
      NR=NA(1)
      NC=NA(2)
      L=NR*NC
      IF( NR .LT. 1 .OR. NC .LT. 1  )  GO TO 999
      NB(1)=NC
      NB(2)=NR
      L=NR*NC
      IR=0
      DO 300 I=1,NR
      IJ=I-NR
      DO 300 J=1,NC
      IJ=IJ+NR
      IR=IR+1
  300 B(IR)=A(IJ)
      RETURN
  999 CONTINUE
      WRITE (MSG,50) NA
      CALL ORCERP(3,MSG)
   50 FORMAT  ('DIMENSION ERROR IN TRANP   NA=',2(1X,I5))
      RETURN
      END
      SUBROUTINE UNITY(A,NA)
      INTEGER
     + NA(2)
      REAL
     + A(*)
      CHARACTER*42
     + MSG
      INTEGER
     + I, IT, J, L, NAX
      IF(NA(1).NE.NA(2) .OR. NA(1).LT.1) GO TO 999
      L=NA(1)*NA(2)
      DO 100 IT=1,L
  100 A(IT) = 0.0E+00
      J = - NA(1)
      NAX = NA(1)
      DO 300 I=1,NAX
      J=NAX +J+1
  300 A(J) = 1.0E+00
      RETURN
  999 CONTINUE
      WRITE (MSG,50) NA
      CALL ORCERP(3,MSG)
   50 FORMAT  ('DIMENSION ERROR IN UNITY   NA=',2(1X,I5))
      END
      SUBROUTINE NULL(A,NA)
      INTEGER
     + NA(2)
      REAL
     + A(*)
      CHARACTER*42
     + MSG
      INTEGER
     + I, N
      IF( NA(1) .LT. 1 .OR.  NA(2) .LT. 1 )  GO TO 999
      N=NA(1)*NA(2)
      DO 10 I=1,N
   10 A(I) = 0.0E+00
      RETURN
  999 CONTINUE
      WRITE (MSG,50) NA
      CALL ORCERP(3,MSG)
   50 FORMAT  ('DIMENSION ERROR IN NULL    NA=',2(1X,I5))
      RETURN
      END
      SUBROUTINE READ1 (A,NA,NZ,NAM)
      CHARACTER*(*)
     + NAM
      INTEGER
     + NA(2), NZ(2)
      REAL
     + A(*)
      COMMON /ORACIO/ NIN, NOUT, NERR
      INTEGER
     + NERR, NIN, NOUT
      SAVE /ORACIO/
      CHARACTER*50
     + MSG
      INTEGER
     + I, J, NC, NLST, NR
      IF  (NZ(1).EQ.0)  GO TO 410
      NR=NZ(1)
      NC=NZ(2)
      NLST=NR*NC
      IF( NLST .LT. 1 .OR. NR .LT. 1 ) GO TO 999
      DO 400 I = 1, NR
      READ (NIN,101,END=998) (A(  J), J = I,NLST,NR)
  400 CONTINUE
      NA(1)=NR
      NA(2)=NC
  410 CALL  PRNT (A,NA,NAM,1)
  101 FORMAT(BZ,8E10.2)
      RETURN
  999 CONTINUE
      WRITE (MSG,916)  NAM,NR,NC
      CALL ORCERP(3,MSG)
  916 FORMAT('ERROR IN READ1   MATRIX ',A4,' HAS NA=',2(1X,I5))
      RETURN
  998 CONTINUE
      WRITE (MSG,917)  NAM
      CALL ORCERP(3,MSG)
  917 FORMAT('ERROR IN READ1, END OF FILE READING MATRIX ',A4)
      RETURN
      END
      SUBROUTINE EXTRCT(A,NA,B,NB,IUL,JUL)
      INTEGER
     + IUL, JUL, NA(2), NB(2)
      REAL
     + A(*), B(*)
C
C     THIS SUBROUTINE EXTRACTS A SUBMATRIX FROM MATRIX  A  AND PLACES IT
C        IN  B.  DATA MAY BE STORED IN  A  IN EITHER PACKED (ORACLS)
C        FORMAT OR IN FORTRAN FORMAT.  DATA ARE PLACED IN  B  IN PACKED
C        (ORACLS) FORMAT
C
C     INPUT PARAMETERS (RETURNED UNCHANGED)
C
C     A - MATRIX WITH DATA STORED EITHER PACKED BY COLUMNS (ORACLS
C        FORMAT) OR AS FORTRAN NATURALLY STORES IT.
C     NA - INTEGER ARRAY OF 2 ENTRIES.  IF  A  IS STORED IN ORACLS
C        FORMAT,  NA(1)  AND  NA(2)  MUST BE SET TO THE ACTUAL ROW AND
C        COLUMN COUNT OF  A.  IF  A  IS IN FORTRAN FORMAT,  NA(1)  MUST
C        BE SET TO THE ROW DIMENSION OF  A  AS DECLARED IN THE DIMENSION
C        STATEMENT IN THE CALLING PROGRAM.  NA(2)  IS USED ONLY IN ERROR
C        CHECKING.  IT SHOULD BE SET TO THE NUMBER OF COLUMNS OF DATA
C        WHICH ARE ACTUALLY IN  A.
C     NB - 2 ENTRY INTEGER ARRAY.  NB(1)  AND  NB(2)  MUST BE SET TO THE
C        ACTUAL ROW AND COLUMN COUNTS OF MATRIX  B.  THIS DEFINES THE
C        SIZE OF THE SUBMATRIX TO BE MOVED.
C     IUL, JUL - INTEGERS SET TO THE ROW AND COLUMN INDICES OF THE UPPER
C        LEFT CORNER OF THE SUBMATRIX OF  A TO BE MOVED.
C
C     OUTPUT PARAMETERS
C
C     B - MATRIX IN WHICH THE DESIGNATED SUBMATRIX OF  A  IS STORED IN
C        ORACLES FORMAT.
C
C     NOTES -
C        (A)  EXTRCT CAN BE USED TO PACK A FORTRAN ARRAY FOR USE BY
C        ORACLS BY SETTING  IUL = JUL = 1,  AND ENTERING IN  NB  THE
C        NUMBER OF ROWS AND COLUMNS OF ACTUAL DATA IN  A  TO BE PACKED.
C        (B)  THE ARRAYS  A  AND  B  CAN OCCUPY NONOVERLAPPING AREAS OF
C        STORAGE OR CAN BE THE SAME.  IF EXTRCT IS USED ON ARRAYS  A
C        AND  B  WHICH OVERLAP WITH OFFSET STARTING POINTS, RESULTS MAY
C        BE WRONG.
C
      CHARACTER*42
     + MSG
      INTEGER
     + I, IA, J, MA, MB, NA1, NA2, NB1, NB2
C
      NA1 = NA(1)
      NA2 = NA(2)
      NB1 = NB(1)
      NB2 = NB(2)
      IF (IUL .LT. 1) GO TO 100
      IF (NB1 .LT. 1) GO TO 100
      IF (NA1 .LT. IUL + NB1 - 1) GO TO 100
      IF (JUL .LT. 1) GO TO 100
      IF (NB2 .LT. 1) GO TO 100
      IF (NA2 .LT. JUL + NB2 - 1) GO TO 100
      MA=(JUL-1)*NA1+IUL
      MB=1
      IA=NA1-NB1
      DO 20 I=1,NB2
      DO 10 J=1,NB1
      B(MB)=A(MA)
      MB=MB+1
      MA=MA+1
   10 CONTINUE
      MA=MA+IA
   20 CONTINUE
      RETURN
  100 CONTINUE
      CALL ORCERP(1,
     + 'DIMENSION AND/OR INDEXING ERROR IN SUBROUTINE EXTRCT')
      WRITE(MSG,110) NA
  110 FORMAT(3X,5HNA = ,2I20)
      CALL ORCERP(0,MSG)
      WRITE(MSG,120) NB
  120 FORMAT(3X,5HNB = ,2I20)
      CALL ORCERP(0,MSG)
      WRITE (MSG,130) IUL,JUL
  130 FORMAT(3X,6HIUL = ,I20,8H, JUL = ,I20)
      CALL ORCERP(2,MSG)
      RETURN
      END
      SUBROUTINE MERGE(B,NB,A,NA,IUL,JUL)
      INTEGER
     + IUL, JUL, NA(2), NB(2)
      REAL
     + A(*), B(*)
C
C     THIS SUBROUTINE WRITES MATRIX  B  OVER A SUBMATRIX OF MATRIX  A.
C        DATA MAY BE STORED IN  A  IN EITHER PACKED (ORACLS) FORMAT OR
C        IN FORTRAN FORMAT.  DATA IN  B  MUST BE IN PACKED  (ORACLS)
C        FORMAT
C
C     INPUT PARAMETERS (RETURNED UNCHANGED)
C
C     B - MATRIX WITH DATA STORED PACKED BY COLUMNS (ORACLS FORMAT).
C     NB - INTEGER ARRAY OF 2 ENTRIES.  NB(1)  AND  NB(2)  MUST BE SET
C        TO THE ACTUAL ROW AND COLUMN COUNT OF  B.
C     NA - INTEGER ARRAY OF 2 ENTRIES.  IF  A  IS IN ORACLS
C        FORMAT,  NA(1)  AND  NA(2)  MUST BE SET TO THE ACTUAL ROW AND
C        COLUMN COUNT OF  A.  IF  A  IS IN FORTRAN FORMAT,  NA(1)
C        AND  NA(2)  MUST BE SET TO THE ROW AND COLUMN DIMENSIONS OF  A
C        AS DECLARED IN THE DIMENSION STATEMENT IN THE CALLING PROGRAM.
C     IUL, JUL - INTEGERS SET TO THE ROW AND COLUMN INDICES OF THE
C        UPPER LEFT CORNER OF THE SUBMATRIX OF  A  TO BE OVERWRITTEN;
C        I.E., THE MATRIX  B  WILL BE MOVED ONTO THE SUBMATRIX OF  A
C        WITH UPPER LEFT CORNER A(IUL,JUL).
C
C     OUTPUT PARAMETER
C
C     A - MATRIX IN WHICH THE DESIGNATED SUBMATRIX WILL BE REPLACED BY
C        THE CONTENTS OF MATRIX  B.  OTHER ELEMENTS OF  A  ARE NOT
C        AFFECTED.
C
C     NOTES -
C        (A)  MERGE CAN BE USED TO UNPACK AN ORACLS ARRAY FOR USE BY
C        FORTRAN BY SETTING  IUL = JUL = 1.
C        (B)  THE ARRAYS  A  AND  B  CAN OCCUPY NONOVERLAPPING AREAS OF
C        STORAGE OR CAN BE THE SAME.  IF MERGE IS USED ON ARRAYS  A  AND
C        B  WHICH OVERLAP WITH OFFSET STARTING POINTS, RESULTS MAY BE
C        WRONG.
C
      CHARACTER*60
     + MSG
      INTEGER
     + I, IA, J, MA, MB, NA1, NA2, NB1, NB2
      NA1=NA(1)
      NA2=NA(2)
      NB1=NB(1)
      NB2=NB(2)
      IF (IUL .LT. 1) GO TO 100
      IF (NB1 .LT. 1) GO TO 100
      IF (NA1 .LT. IUL+NB1-1) GO TO 100
      IF (JUL .LT. 1) GO TO 100
      IF (NB2 .LT. 1) GO TO 100
      IF (NA2 .LT. JUL+NB2-1) GO TO 100
      MA=(JUL+NB2-2)*NA1+IUL+NB1-1
      MB=NB1*NB2
      IA=NA1-NB1
      DO 20 I=1,NB2
      DO 10 J=1,NB1
      A(MA)=B(MB)
      MB=MB-1
      MA=MA-1
   10 CONTINUE
      MA=MA-IA
   20 CONTINUE
      RETURN
  100 CONTINUE
      CALL LNCNT(6)
      CALL ORCERP(1,
     + 'DIMENSION AND/OR INDEXING ERROR IN SUBROUTINE MERGE')
      WRITE(MSG,110) NA
  110 FORMAT(3X,5HNA = ,2I20)
      CALL ORCERP(0,MSG)
      WRITE(MSG,120) NB
  120 FORMAT(3X,5HNB = ,2I20)
      CALL ORCERP(0,MSG)
      WRITE (MSG,130) IUL,JUL
  130 FORMAT(3X,6HIUL = ,I20,8H, JUL = ,I20)
      CALL ORCERP(2,MSG)
      RETURN
      END
```

Appendix E

Output From Example Computation

The following pages give the output of the example program as described in the section "Example" and listed in appendix D.

The format shown was achieved by setting the page length parameter (NLP) to 55, the line length parameter (NCL) to 80, and the page title parameter (TITLE) to

```
'DEMONSTRATION OF FREQ, BASED ON EXAMPLE IN NASA TP-2560'
```

Column 1 of the output file contains carriage control information ('1' to indicate the first line of a new page, ' ' to print the line after a single-line space advance) and is not printed here; instead, the effect of using the carriage control information is shown.

```
DEMONSTRATION OF FREQ, BASED ON EXAMPLE IN NASA TP-2560
                                                            FREQ    PROGRAM
DATA FROM NASA TP-2560 USED TO GENERATE LINEAR SYSTEM MATRICES FOR
    THE 26TH-ORDER PLANT
    THE 12TH-ORDER DESIGN MODEL
    THE 14TH-ORDER ERROR (DELTA G) SYSTEM
    THE 12TH-ORDER ATTITUDE FEEDBACK COMPENSATOR

    "FNDR" CONTAINS THE FREQUENCIES (HZ) AND THE DAMPING RATIOS
    "VB" CONTAINS MODE SLOPE INFORMATION FOR "B" AND "C" MATRICES
    "H" IS THE KALMAN FILTER GAINS MATRIX
    "GT" IS THE TRANSPOSE OF THE FEEDBACK GAINS MATRIX

    FNDR  MATRIX      10 ROWS       2 COLUMNS
 7.5000E-01   1.0000E-02
 1.3500E+00   1.0000E-02
 1.7000E+00   1.0000E-02
 3.1800E+00   1.0000E-02
 4.5300E+00   1.0000E-02
 5.5900E+00   1.0000E-02
 5.7800E+00   1.0000E-02
 6.8400E+00   1.0000E-02
 7.4000E+00   1.0000E-02
 8.7800E+00   1.0000E-02

    RINV  MATRIX       3 ROWS       3 COLUMNS
 2.3700E-07    .0000E+00    .0000E+00
  .0000E+00   2.3700E-07    .0000E+00
  .0000E+00    .0000E+00   3.0900E-07

    VB    MATRIX      20 ROWS       3 COLUMNS
  .0000E+00    .0000E+00    .0000E+00
 6.3000E-05    .0000E+00  -5.0400E-03
  .0000E+00    .0000E+00    .0000E+00
  .0000E+00  -2.9100E-03    .0000E+00
  .0000E+00    .0000E+00    .0000E+00
 3.7100E-03    .0000E+00   5.4500E-04
  .0000E+00    .0000E+00    .0000E+00
-7.4900E-07    .0000E+00  -8.4500E-06
  .0000E+00    .0000E+00    .0000E+00
-6.2100E-04    .0000E+00  -5.6000E-05
  .0000E+00    .0000E+00    .0000E+00
 5.5500E-06    .0000E+00  -3.1900E-04
  .0000E+00    .0000E+00    .0000E+00
  .0000E+00  -5.0000E-04    .0000E+00
  .0000E+00    .0000E+00    .0000E+00
 1.1000E-03    .0000E+00  -2.2300E-04
  .0000E+00    .0000E+00    .0000E+00
  .0000E+00  -3.0400E-03    .0000E+00
  .0000E+00    .0000E+00    .0000E+00
 2.4300E-03    .0000E+00  -1.0500E-04

    H     MATRIX      12 ROWS       3 COLUMNS
```

```
DEMONSTRATION OF FREQ, BASED ON EXAMPLE IN NASA TP-2560
```

```
  4.4700E-01  -9.3200E-09   4.3600E-08
 -2.1200E-08   4.4700E-01   7.7200E-08
 -2.4400E-08   1.4800E-07   4.4700E-01
  1.0000E-01   1.3300E-09   7.6000E-09
 -1.3300E-09   1.0000E-01  -7.9600E-09
 -7.6000E-09   7.9700E-09   1.0000E-01
 -1.3900E-05   1.4200E-05  -3.6000E-09
 -7.9200E-06  -1.4100E-05  -1.9400E-09
 -5.1200E-06   1.4300E-06  -3.3400E-11
 -2.5900E-06  -5.1600E-06   1.1900E-10
 -3.2900E-06   5.6900E-07   8.9200E-11
 -1.6500E-06  -3.3100E-06   1.2600E-10

   GT    MATRIX      12 ROWS       3 COLUMNS
  1.0000E+05  -2.7300E-07  -7.6100E+02
 -9.8500E-07   1.0000E+05  -6.2000E-07
  7.6100E+02   4.5000E-07   1.0000E+05
  7.5200E+05   2.2600E-05  -8.6000E+03
  1.2300E-05   7.6100E+05  -7.4400E-06
 -3.3200E+02   3.6400E-06   3.3500E+05
 -2.6300E+00  -1.9100E-10  -4.1200E+02
  9.5500E+00   2.4900E-10  -3.8300E+02
  2.5700E-08  -9.3100E+01   3.6700E-09
  2.8800E-08  -1.9500E+02   1.0000E-09
  1.1400E+02  -6.8100E-09   3.6000E+01
  1.9800E+02   1.4130E-09   2.3200E+01

NEXT CALCULATE THE OPEN-LOOP RESPONSE OF THE FULL 26TH-ORDER PLANT
```

```
 PROGRAM TO COMPUTE MATRIX G EVALUATED AT
 S = SQRT(-1)XOMEGA, OMEGA =      .1000000E-01

 GO = CG( (SI - AG)INVERSE )BG + DG

     CG   MATRIX        3 ROWS       26 COLUMNS
  1.0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   6.3000E-05    .0000E+00    .0000E+00    .0000E+00   3.7100E-03    .0000E+00
  -7.4900E-07    .0000E+00  -6.2100E-04    .0000E+00   5.5500E-06    .0000E+00
    .0000E+00    .0000E+00   1.1000E-03    .0000E+00    .0000E+00    .0000E+00
   2.4300E-03    .0000E+00
   .0000E+00   1.0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00  -2.9100E-03    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
  -5.0000E-04    .0000E+00    .0000E+00    .0000E+00  -3.0400E-03    .0000E+00
    .0000E+00    .0000E+00
   .0000E+00    .0000E+00   1.0000E+00    .0000E+00    .0000E+00    .0000E+00
  -5.0400E-03    .0000E+00    .0000E+00    .0000E+00   5.4500E-04    .0000E+00
  -8.4500E-06    .0000E+00  -5.6000E-05    .0000E+00  -3.1900E-04    .0000E+00
    .0000E+00    .0000E+00  -2.2300E-04    .0000E+00    .0000E+00    .0000E+00
  -1.0500E-04    .0000E+00

     AG   MATRIX       26 ROWS       26 COLUMNS
  .0000E+00    .0000E+00    .0000E+00   1.0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00   1.0000E+00    .0000E+00
    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00   1.0000E+00
    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
    .0000E+00    .0000E+00
```

```
DEMONSTRATION OF FREQ, BASED ON EXAMPLE IN NASA TP-2560
  .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00
  .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00    1.0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00
  .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
 -5.6250E-01   -1.5000E-02     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00
  .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00    1.0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00
  .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00   -1.8225E+00   -2.7000E-02     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00
  .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00    1.0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00
  .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00   -2.8900E+00   -3.4000E-02
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00
  .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00    1.0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00
  .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
 -1.0112E+01   -6.3600E-02     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00
  .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00    1.0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
   .0000E+00     .0000E+00
```

```
DEMONSTRATION OF FREQ, BASED ON EXAMPLE IN NASA TP-2560
                                                                      FREQ    PROGRAM
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00  -2.0521E+01  -9.0600E-02    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00   1.0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00  -3.1248E+01  -1.1180E-01
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00   1.0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
 -3.3408E+01  -1.1560E-01    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00   1.0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00  -4.6786E+01  -1.3680E-01    .0000E+00    .0000E+00
   .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00   1.0000E+00
   .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00  -5.4760E+01  -1.4800E-01
   .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00   1.0000E+00
```

```
DEMONSTRATION OF FREQ, BASED ON EXAMPLE IN NASA TP-2560
                                                                FREQ    PROGRAM
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
 -7.7088E+01  -1.7560E-01

     BG   MATRIX      26 ROWS       3 COLUMNS
   .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00
  2.3700E-07    .0000E+00    .0000E+00
   .0000E+00   2.3700E-07    .0000E+00
   .0000E+00    .0000E+00   3.0900E-07
   .0000E+00    .0000E+00    .0000E+00
  6.3000E-05    .0000E+00  -5.0400E-03
   .0000E+00    .0000E+00    .0000E+00
   .0000E+00  -2.9100E-03    .0000E+00
   .0000E+00    .0000E+00    .0000E+00
  3.7100E-03    .0000E+00   5.4500E-04
   .0000E+00    .0000E+00    .0000E+00
 -7.4900E-07    .0000E+00  -8.4500E-06
   .0000E+00    .0000E+00    .0000E+00
 -6.2100E-04    .0000E+00  -5.6000E-05
   .0000E+00    .0000E+00    .0000E+00
  5.5500E-06    .0000E+00  -3.1900E-04
   .0000E+00    .0000E+00    .0000E+00
   .0000E+00  -5.0000E-04    .0000E+00
   .0000E+00    .0000E+00    .0000E+00
  1.1000E-03    .0000E+00  -2.2300E-04
   .0000E+00    .0000E+00    .0000E+00
   .0000E+00  -3.0400E-03    .0000E+00
   .0000E+00    .0000E+00    .0000E+00
  2.4300E-03    .0000E+00  -1.0500E-04
 DG IS A NULL MATRIX

 G = GO, LOOP-GAIN TRANSFER MATRIX
 SINGULAR VALUES BUT NO SINGULAR VECTORS OF G ARE COMPUTED

 SYSTEM MATRIX IS INPUT AG

 RCOND =    6.1791092E-06
 RCOND = ESTIMATE OF RECIPROCAL OF CONDITION NUMBER FOR
        H = SQRT(-1)*OMEGA*I - SYSTEM MATRIX
```

```
DEMONSTRATION OF FREQ, BASED ON EXAMPLE IN NASA TP-2560
                                                              FREQ    PROGRAM

 EIGENVALUES, (REAL PART,IMAG. PART), OF SYSTEM MATRIX
 (      -6.840000E-02 ,      6.839658E+00 )
 (      -6.840000E-02 ,     -6.839658E+00 )
 (      -7.400000E-02 ,      7.399630E+00 )
 (      -7.400000E-02 ,     -7.399630E+00 )
 (      -8.780000E-02 ,      8.779561E+00 )
 (      -8.780000E-02 ,     -8.779561E+00 )
 (      -7.500000E-03 ,      7.499625E-01 )
 (      -7.500000E-03 ,     -7.499625E-01 )
 (      -1.350000E-02 ,      1.349932E+00 )
 (      -1.350000E-02 ,     -1.349932E+00 )
 (      -1.700000E-02 ,      1.699915E+00 )
 (      -1.700000E-02 ,     -1.699915E+00 )
 (      -3.180000E-02 ,      3.179841E+00 )
 (      -3.180000E-02 ,     -3.179841E+00 )
 (      -4.530000E-02 ,      4.529773E+00 )
 (      -4.530000E-02 ,     -4.529773E+00 )
 (      -5.590000E-02 ,      5.589720E+00 )
 (      -5.590000E-02 ,     -5.589720E+00 )
 (      -5.780000E-02 ,      5.779711E+00 )
 (      -5.780000E-02 ,     -5.779711E+00 )
 (        .000000E+00 ,       .000000E+00 )
 (        .000000E+00 ,       .000000E+00 )
 (        .000000E+00 ,       .000000E+00 )
 (        .000000E+00 ,       .000000E+00 )
 (        .000000E+00 ,       .000000E+00 )
 (        .000000E+00 ,       .000000E+00 )

    G    COMPLEX MATRIX       3 ROWS       3 COLUMNS
(-2.3651E-03,-5.6557E-10) (  .0000E+00,   .0000E+00) ( 1.2817E-07, 6.8422E-11)
(  .0000E+00,   .0000E+00) (-2.3652E-03,-6.9325E-10) (  .0000E+00,   .0000E+00)
( 1.2817E-07, 6.8422E-11) (  .0000E+00,   .0000E+00) (-3.0447E-03,-1.2059E-08)

 VECTOR SIGMA OF COMPUTED SINGULAR VALUES OF G

 SIGMA(   1) =     .3044726E-02
 SIGMA(   2) =     .2365177E-02
 SIGMA(   3) =     .2365109E-02

NEXT CALCULATE THE OPEN-LOOP RESPONSE OF THE 12TH-ORDER DESIGN MODEL
```

```
PROGRAM TO COMPUTE MATRIX G EVALUATED AT
S = SQRT(-1)XOMEGA, OMEGA =      .1000000E-01

GO = CG( (SI - AG)INVERSE )BG + DG

     CG   MATRIX       3 ROWS      12 COLUMNS
 1.0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
  6.3000E-05    .0000E+00    .0000E+00    .0000E+00   3.7100E-03    .0000E+00
  .0000E+00   1.0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00  -2.9100E-03    .0000E+00    .0000E+00    .0000E+00
  .0000E+00    .0000E+00   1.0000E+00    .0000E+00    .0000E+00    .0000E+00
 -5.0400E-03    .0000E+00    .0000E+00    .0000E+00   5.4500E-04    .0000E+00

     AG   MATRIX      12 ROWS      12 COLUMNS
  .0000E+00    .0000E+00    .0000E+00   1.0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00   1.0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00   1.0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00   1.0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
 -5.6250E-01  -1.5000E-02    .0000E+00    .0000E+00    .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00   1.0000E+00    .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00  -1.8225E+00  -2.7000E-02    .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00   1.0000E+00
  .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00    .0000E+00
   .0000E+00    .0000E+00    .0000E+00    .0000E+00  -2.8900E+00  -3.4000E-02

     BG   MATRIX      12 ROWS       3 COLUMNS
  .0000E+00    .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00
  .0000E+00    .0000E+00    .0000E+00
 2.3700E-07    .0000E+00    .0000E+00
  .0000E+00   2.3700E-07    .0000E+00
  .0000E+00    .0000E+00   3.0900E-07
  .0000E+00    .0000E+00    .0000E+00
 6.3000E-05    .0000E+00  -5.0400E-03
  .0000E+00    .0000E+00    .0000E+00
```

```
DEMONSTRATION OF FREQ, BASED ON EXAMPLE IN NASA TP-2560

   .0000E+00  -2.9100E-03    .0000E+00
   .0000E+00    .0000E+00    .0000E+00
  3.7100E-03    .0000E+00   5.4500E-04

 DG IS A NULL MATRIX

 G = GO, LOOP-GAIN TRANSFER MATRIX

 SINGULAR VALUES BUT NO SINGULAR VECTORS OF G ARE COMPUTED

 SYSTEM MATRIX IS INPUT AG

 RCOND =     3.4482494E-05
 RCOND = ESTIMATE OF RECIPROCAL OF CONDITION NUMBER FOR
         H = SQRT(-1)*OMEGA*I - SYSTEM MATRIX

 EIGENVALUES, (REAL PART,IMAG. PART), OF SYSTEM MATRIX

 (     -7.500000E-03 ,      7.499625E-01 )
 (     -7.500000E-03 ,     -7.499625E-01 )
 (     -1.350000E-02 ,      1.349932E+00 )
 (     -1.350000E-02 ,     -1.349932E+00 )
 (     -1.700000E-02 ,      1.699915E+00 )
 (     -1.700000E-02 ,     -1.699915E+00 )
 (       .000000E+00 ,       .000000E+00 )
 (       .000000E+00 ,       .000000E+00 )
 (       .000000E+00 ,       .000000E+00 )
 (       .000000E+00 ,       .000000E+00 )
 (       .000000E+00 ,       .000000E+00 )
 (       .000000E+00 ,       .000000E+00 )

    G    COMPLEX MATRIX       3 ROWS       3 COLUMNS
(-2.3652E-03,-5.6223E-10) (  .0000E+00,  .0000E+00) ( 1.3508E-07, 6.8266E-11)
(  .0000E+00,  .0000E+00) (-2.3654E-03,-6.8843E-10) (  .0000E+00,  .0000E+00)
( 1.3508E-07, 6.8266E-11) (  .0000E+00,  .0000E+00) (-3.0447E-03,-1.2059E-08)

 VECTOR SIGMA OF COMPUTED SINGULAR VALUES OF G

 SIGMA(   1) =     .3044731E-02
 SIGMA(   2) =     .2365353E-02
 SIGMA(   3) =     .2365230E-02

NEXT CALCULATE THE SINGULAR VALUE PLOTS OF THE 14TH-ORDER ERROR SYSTEM
```

```
DEMONSTRATION OF FREQ, BASED ON EXAMPLE IN NASA TP-2560
                                                              FREQ    PROGRAM

 PROGRAM TO COMPUTE MATRIX G EVALUATED AT
 S = SQRT(-1)XOMEGA, OMEGA =     .1000000E-01

 GO = CG( (SI - AG)INVERSE )BG + DG

      CG   MATRIX       3 ROWS      14 COLUMNS
 -7.4900E-07   .0000E+00  -6.2100E-04   .0000E+00   5.5500E-06   .0000E+00
    .0000E+00   .0000E+00   1.1000E-03   .0000E+00   .0000E+00   .0000E+00
   2.4300E-03   .0000E+00
   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
  -5.0000E-04   .0000E+00   .0000E+00   .0000E+00  -3.0400E-03   .0000E+00
    .0000E+00   .0000E+00
 -8.4500E-06   .0000E+00  -5.6000E-05   .0000E+00  -3.1900E-04   .0000E+00
    .0000E+00   .0000E+00  -2.2300E-04   .0000E+00   .0000E+00   .0000E+00
  -1.0500E-04   .0000E+00

      AG   MATRIX      14 ROWS      14 COLUMNS
   .0000E+00   1.0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00
 -1.0112E+01  -6.3600E-02   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00   1.0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00
   .0000E+00   .0000E+00  -2.0521E+01  -9.0600E-02   .0000E+00   .0000E+00
    .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   1.0000E+00
    .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00   .0000E+00  -3.1248E+01  -1.1180E-01
    .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   1.0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
  -3.3408E+01  -1.1560E-01   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00   .0000E+00   1.0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00  -4.6786E+01  -1.3680E-01   .0000E+00   .0000E+00
    .0000E+00   .0000E+00
```

```
DEMONSTRATION OF FREQ, BASED ON EXAMPLE IN NASA TP-2560
                                                                        FREQ    PROGRAM
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
    .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00    1.0000E+00
    .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
    .0000E+00     .0000E+00     .0000E+00     .0000E+00   -5.4760E+01   -1.4800E-01
    .0000E+00     .0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
    .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
    .0000E+00    1.0000E+00
   .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
    .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00     .0000E+00
  -7.7088E+01   -1.7560E-01

      BG   MATRIX      14 ROWS       3 COLUMNS
   .0000E+00     .0000E+00     .0000E+00
 -7.4900E-07     .0000E+00   -8.4500E-06
   .0000E+00     .0000E+00     .0000E+00
 -6.2100E-04     .0000E+00   -5.6000E-05
   .0000E+00     .0000E+00     .0000E+00
  5.5500E-06     .0000E+00   -3.1900E-04
   .0000E+00     .0000E+00     .0000E+00
   .0000E+00   -5.0000E-04     .0000E+00
   .0000E+00     .0000E+00     .0000E+00
  1.1000E-03     .0000E+00   -2.2300E-04
   .0000E+00     .0000E+00     .0000E+00
   .0000E+00   -3.0400E-03     .0000E+00
   .0000E+00     .0000E+00     .0000E+00
  2.4300E-03     .0000E+00   -1.0500E-04

 DG IS A NULL MATRIX

 G = GO, LOOP-GAIN TRANSFER MATRIX

 SINGULAR VALUES BUT NO SINGULAR VECTORS OF G ARE COMPUTED

SYSTEM MATRIX IS INPUT AG

 RCOND =     1.0618094E-01
 RCOND = ESTIMATE OF RECIPROCAL OF CONDITION NUMBER FOR
         H = SQRT(-1)*OMEGA*I - SYSTEM MATRIX

 EIGENVALUES, (REAL PART,IMAG. PART), OF SYSTEM MATRIX

 (     -3.180000E-02 ,       3.179841E+00 )
 (     -3.180000E-02 ,      -3.179841E+00 )
 (     -4.530000E-02 ,       4.529773E+00 )
```

```
DEMONSTRATION OF FREQ, BASED ON EXAMPLE IN NASA TP-2560
                                                            FREQ    PROGRAM
 (     -4.530000E-02 ,     -4.529773E+00 )
 (     -5.590000E-02 ,      5.589720E+00 )
 (     -5.590000E-02 ,     -5.589720E+00 )
 (     -5.780000E-02 ,      5.779711E+00 )
 (     -5.780000E-02 ,     -5.779711E+00 )
 (     -6.840000E-02 ,      6.839658E+00 )
 (     -6.840000E-02 ,     -6.839658E+00 )
 (     -7.400000E-02 ,      7.399630E+00 )
 (     -7.400000E-02 ,     -7.399630E+00 )
 (     -8.780000E-02 ,      8.779561E+00 )
 (     -8.780000E-02 ,     -8.779561E+00 )

    G    COMPLEX MATRIX       3 ROWS       3 COLUMNS
( 1.2126E-07,-3.3308E-12) (  .0000E+00,  .0000E+00) (-6.9143E-09, 1.5587E-13)
(  .0000E+00,  .0000E+00) ( 1.7625E-07,-4.8202E-12) (  .0000E+00,  .0000E+00)
(-6.9143E-09, 1.5587E-13) (  .0000E+00,  .0000E+00) ( 4.6224E-09,-1.5804E-13)

 VECTOR SIGMA OF COMPUTED SINGULAR VALUES OF G

 SIGMA(  1) =     .1762490E-06
 SIGMA(  2) =     .1216641E-06
 SIGMA(  3) =     .4213911E-08

NEXT CALCULATE THE OPEN-LOOP RESPONSE OF THE COMPENSATOR
```

```
DEMONSTRATION OF FREQ, BASED ON EXAMPLE IN NASA TP-2560
                                                                  FREQ    PROGRAM

 PROGRAM TO COMPUTE MATRIX G EVALUATED AT
 S = SQRT(-1)XOMEGA, OMEGA =     .1000000E-01
 GO = CG( (SI - AG)INVERSE )BG + DG

     CG   MATRIX       3 ROWS      12 COLUMNS
  1.0000E+05  -9.8500E-07   7.6100E+02   7.5200E+05   1.2300E-05  -3.3200E+02
  -2.6300E+00   9.5500E+00   2.5700E-08   2.8800E-08   1.1400E+02   1.9800E+02
 -2.7300E-07   1.0000E+05   4.5000E-07   2.2600E-05   7.6100E+05   3.6400E-06
  -1.9100E-10   2.4900E-10  -9.3100E+01  -1.9500E+02  -6.8100E-09   1.4130E-09
 -7.6100E+02  -6.2000E-07   1.0000E+05  -8.6000E+03  -7.4400E-06   3.3500E+05
  -4.1200E+02  -3.8300E+02   3.6700E-09   1.0000E-09   3.6000E+01   2.3200E+01

     AG   MATRIX      12 ROWS      12 COLUMNS
 -4.4700E-01   9.3200E-09  -4.3600E-08   1.0000E+00    .0000E+00    .0000E+00
  -2.8161E-05    .0000E+00  -2.7121E-11    .0000E+00  -1.6584E-03    .0000E+00
  2.1200E-08  -4.4700E-01  -7.7200E-08    .0000E+00   1.0000E+00    .0000E+00
   3.9042E-10    .0000E+00   1.3008E-03    .0000E+00   3.6578E-11    .0000E+00
  2.4400E-08  -1.4800E-07  -4.4700E-01    .0000E+00    .0000E+00   1.0000E+00
   2.2529E-03    .0000E+00   4.3068E-10    .0000E+00  -2.4361E-04    .0000E+00
 -1.2370E-01  -1.3298E-09  -1.8036E-04  -1.7822E-01  -2.9151E-12   7.8684E-05
  -5.6767E-06  -2.2634E-06   3.8642E-12  -6.8256E-15  -3.9802E-04  -4.6926E-05
  1.3301E-09  -1.2370E-01   7.9599E-09  -5.3562E-12  -1.8036E-01  -8.6268E-13
  -4.0035E-11  -5.9013E-17   3.1306E-04   4.6215E-05   9.2741E-12  -3.3488E-16
  2.3516E-04  -7.9698E-09  -1.3090E-01   2.6574E-03   2.2990E-12  -1.0352E-01
   6.3131E-04   1.1835E-04   2.3192E-11  -3.0900E-16  -6.5624E-05  -7.1688E-06
  1.3900E-05  -1.4200E-05   3.6000E-09    .0000E+00    .0000E+00    .0000E+00
   8.5756E-10   1.0000E+00   4.1322E-08    .0000E+00   5.1571E-08    .0000E+00
 -1.0135E+01   1.4097E-05   5.0395E+02  -9.0720E+01  -3.8273E-08   1.6884E+03
  -2.6388E+00  -1.9459E+00  -4.1014E-08   3.2256E-12   1.7426E-01   1.0445E-01
  5.1200E-06  -1.4300E-06   3.3400E-11    .0000E+00    .0000E+00    .0000E+00
   3.2239E-10    .0000E+00   4.1613E-09   1.0000E+00   1.8995E-08    .0000E+00
  2.5892E-06   2.9100E+02   1.1905E-09   6.5766E-08   2.2145E+03   1.0592E-08
   1.6321E-10   7.2459E-13  -2.0934E+00  -5.9445E-01   9.5890E-09   4.1118E-12
  3.2900E-06  -5.6900E-07  -8.9200E-11    .0000E+00    .0000E+00    .0000E+00
   2.0772E-10    .0000E+00   1.6558E-09    .0000E+00   1.2206E-08   1.0000E+00
 -3.7059E+02   3.3140E-06  -5.7323E+01  -2.7852E+03  -4.1578E-08  -1.8134E+02
   2.3430E-01   1.7330E-01  -9.7294E-09  -1.0739E-10  -3.3326E+00  -7.8122E-01

     BG   MATRIX      12 ROWS       3 COLUMNS
  4.4700E-01  -9.3200E-09   4.3600E-08
 -2.1200E-08   4.4700E-01   7.7200E-08
 -2.4400E-08   1.4800E-07   4.4700E-01
  1.0000E-01   1.3300E-09   7.6000E-09
 -1.3300E-09   1.0000E-01  -7.9600E-09
 -7.6000E-09   7.9700E-09   1.0000E-01
 -1.3900E-05   1.4200E-05  -3.6000E-09
 -7.9200E-06  -1.4100E-05  -1.9400E-09
 -5.1200E-06   1.4300E-06  -3.3400E-11
```

```
DEMONSTRATION OF FREQ, BASED ON EXAMPLE IN NASA TP-2560
                                                                   FREQ    PROGRAM
 -2.5900E-06  -5.1600E-06   1.1900E-10
 -3.2900E-06   5.6900E-07   8.9200E-11
 -1.6500E-06  -3.3100E-06   1.2600E-10

 DG IS A NULL MATRIX

 G = GO, LOOP-GAIN TRANSFER MATRIX

 SINGULAR VALUES BUT NO SINGULAR VECTORS OF G ARE COMPUTED

 SYSTEM MATRIX IS INPUT AG

 RCOND =    6.4820193E-03
 RCOND = ESTIMATE OF RECIPROCAL OF CONDITION NUMBER FOR
         H = SQRT(-1)*OMEGA*I - SYSTEM MATRIX

 EIGENVALUES, (REAL PART,IMAG. PART), OF SYSTEM MATRIX

 (     -6.699942E-01 ,      1.781728E+00 )
 (     -6.699942E-01 ,     -1.781728E+00 )
 (     -1.320700E+00 ,      1.529557E+00 )
 (     -1.320700E+00 ,     -1.529557E+00 )
 (     -5.792836E-01 ,      1.423450E+00 )
 (     -5.792836E-01 ,     -1.423450E+00 )
 (      7.261809E-02 ,      8.830831E-02 )
 (      7.261809E-02 ,     -8.830831E-02 )
 (     -3.336587E-02 ,      3.660153E-01 )
 (     -3.336587E-02 ,     -3.660153E-01 )
 (     -3.161989E-02 ,      3.637067E-01 )
 (     -3.161989E-02 ,     -3.637067E-01 )

    G    COMPLEX MATRIX       3 ROWS       3 COLUMNS
( 5.8832E+04, 6.5483E+03) ( 1.0497E-03, 6.3063E-05) ( 4.9397E+02, 1.1730E+02)
( 6.4537E-04,-6.1292E-06) ( 5.8002E+04, 6.4479E+03) (-1.7400E-02,-2.2371E-03)
( 4.7052E+02, 1.1765E+02) (-3.8365E-03,-4.3116E-04) ( 1.0462E+05, 1.9395E+04)

 VECTOR SIGMA OF COMPUTED SINGULAR VALUES OF G

 SIGMA(   1) =     .1064070E+06
 SIGMA(   2) =     .5919041E+05
 SIGMA(   3) =     .5835953E+05
```

```
DEMONSTRATION OF FREQ, BASED ON EXAMPLE IN NASA TP-2560
                                                          FREQ    PROGRAM

NEXT CALCULATE THE OPEN-LOOP RESPONSE OF THE
  CASCADE OF COMPENSATOR AND 12TH-ORDER DESIGN MODEL
```

```
 PROGRAM TO COMPUTE MATRIX G EVALUATED AT
 S = SQRT(-1)XOMEGA, OMEGA =      .1000000E-01

 GO = CG( (SI - AG)INVERSE )BG + DG

      CG   MATRIX        3 ROWS       12 COLUMNS
  1.0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
   6.3000E-05   .0000E+00   .0000E+00   .0000E+00  3.7100E-03   .0000E+00
   .0000E+00  1.0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00 -2.9100E-03   .0000E+00   .0000E+00   .0000E+00
   .0000E+00   .0000E+00  1.0000E+00   .0000E+00   .0000E+00   .0000E+00
  -5.0400E-03   .0000E+00   .0000E+00   .0000E+00  5.4500E-04   .0000E+00

      AG   MATRIX       12 ROWS       12 COLUMNS
   .0000E+00   .0000E+00   .0000E+00  1.0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00   .0000E+00  1.0000E+00   .0000E+00
    .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00  1.0000E+00
    .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00  1.0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
  -5.6250E-01 -1.5000E-02   .0000E+00   .0000E+00   .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00   .0000E+00  1.0000E+00   .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00 -1.8225E+00 -2.7000E-02   .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00  1.0000E+00
   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00   .0000E+00
    .0000E+00   .0000E+00   .0000E+00   .0000E+00 -2.8900E+00 -3.4000E-02

      BG   MATRIX       12 ROWS        3 COLUMNS
   .0000E+00   .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00
   .0000E+00   .0000E+00   .0000E+00
  2.3700E-07   .0000E+00   .0000E+00
   .0000E+00  2.3700E-07   .0000E+00
   .0000E+00   .0000E+00  3.0900E-07
   .0000E+00   .0000E+00   .0000E+00
  6.3000E-05   .0000E+00 -5.0400E-03
  .0000E+00   .0000E+00   .0000E+00
```

```
   .0000E+00  -2.9100E-03   .0000E+00
   .0000E+00   .0000E+00    .0000E+00
  3.7100E-03   .0000E+00   5.4500E-04

 DG IS A NULL MATRIX

 KO = CK( (SI - AK)INVERSE )BK + DK

      CK   MATRIX       3 ROWS      12 COLUMNS
  1.0000E+05  -9.8500E-07   7.6100E+02   7.5200E+05   1.2300E-05  -3.3200E+02
  -2.6300E+00  9.5500E+00   2.5700E-08   2.8800E-08   1.1400E+02   1.9800E+02
 -2.7300E-07   1.0000E+05   4.5000E-07   2.2600E-05   7.6100E+05   3.6400E-06
  -1.9100E-10  2.4900E-10  -9.3100E+01  -1.9500E+02  -6.8100E-09   1.4130E-09
 -7.6100E+02  -6.2000E-07   1.0000E+05  -8.6000E+03  -7.4400E-06   3.3500E+05
  -4.1200E+02 -3.8300E+02   3.6700E-09   1.0000E-09   3.6000E+01   2.3200E+01

     AK   MATRIX       12 ROWS      12 COLUMNS
 -4.4700E-01   9.3200E-09  -4.3600E-08   1.0000E+00    .0000E+00    .0000E+00
  -2.8161E-05   .0000E+00  -2.7121E-11    .0000E+00  -1.6584E-03    .0000E+00
  2.1200E-08  -4.4700E-01  -7.7200E-08    .0000E+00   1.0000E+00    .0000E+00
   3.9042E-10   .0000E+00   1.3008E-03    .0000E+00   3.6578E-11    .0000E+00
  2.4400E-08  -1.4800E-07  -4.4700E-01    .0000E+00    .0000E+00   1.0000E+00
   2.2529E-03   .0000E+00   4.3068E-10    .0000E+00  -2.4361E-04    .0000E+00
 -1.2370E-01  -1.3298E-09  -1.8036E-04  -1.7822E-01  -2.9151E-12   7.8684E-05
  -5.6767E-06 -2.2634E-06   3.8642E-12  -6.8256E-15  -3.9802E-04  -4.6926E-05
  1.3301E-09  -1.2370E-01   7.9599E-09  -5.3562E-12  -1.8036E-01  -8.6268E-13
  -4.0035E-11 -5.9013E-17   3.1306E-04   4.6215E-05   9.2741E-12  -3.3488E-16
  2.3516E-04  -7.9698E-09  -1.3090E-01   2.6574E-03   2.2990E-12  -1.0352E-01
   6.3131E-04  1.1835E-04   2.3192E-11  -3.0900E-16  -6.5624E-05  -7.1688E-06
  1.3900E-05  -1.4200E-05   3.6000E-09    .0000E+00    .0000E+00    .0000E+00
   8.5756E-10  1.0000E+00   4.1322E-08    .0000E+00   5.1571E-08    .0000E+00
 -1.0135E+01   1.4097E-05   5.0395E+02  -9.0720E+01  -3.8273E-08   1.6884E+03
  -2.6388E+00 -1.9459E+00  -4.1014E-08   3.2256E-12   1.7426E-01   1.0445E-01
  5.1200E-06  -1.4300E-06   3.3400E-11    .0000E+00    .0000E+00    .0000E+00
   3.2239E-10   .0000E+00   4.1613E-09   1.0000E+00   1.8995E-08    .0000E+00
  2.5892E-06   2.9100E+02   1.1905E-09   6.5766E-08   2.2145E+03   1.0592E-08
   1.6321E-10  7.2459E-13  -2.0934E+00  -5.9445E-01   9.5890E-09   4.1118E-12
  3.2900E-06  -5.6900E-07  -8.9200E-11    .0000E+00    .0000E+00    .0000E+00
   2.0772E-10   .0000E+00   1.6558E-09    .0000E+00   1.2206E-08   1.0000E+00
 -3.7059E+02   3.3140E-06  -5.7323E+01  -2.7852E+03  -4.1578E-08  -1.8134E+02
   2.3430E-01  1.7330E-01  -9.7294E-09  -1.0739E-10  -3.3326E+00  -7.8122E-01

     BK   MATRIX       12 ROWS       3 COLUMNS
  4.4700E-01  -9.3200E-09   4.3600E-08
 -2.1200E-08   4.4700E-01   7.7200E-08
 -2.4400E-08   1.4800E-07   4.4700E-01
  1.0000E-01   1.3300E-09   7.6000E-09
 -1.3300E-09   1.0000E-01  -7.9600E-09
 -7.6000E-09   7.9700E-09   1.0000E-01
 -1.3900E-05   1.4200E-05  -3.6000E-09
 -7.9200E-06  -1.4100E-05  -1.9400E-09
```

```
DEMONSTRATION OF FREQ, BASED ON EXAMPLE IN NASA TP-2560

 -5.1200E-06   1.4300E-06  -3.3400E-11
 -2.5900E-06  -5.1600E-06   1.1900E-10
 -3.2900E-06   5.6900E-07   8.9200E-11
 -1.6500E-06  -3.3100E-06   1.2600E-10

 DK IS A NULL MATRIX

 GL = GO * KO

 G = GL, LOOP-GAIN TRANSFER MATRIX

 SINGULAR VALUES BUT NO SINGULAR VECTORS OF G ARE COMPUTED

 GO CALCULATION

 RCOND =     3.4482494E-05
 RCOND = ESTIMATE OF RECIPROCAL OF CONDITION NUMBER FOR
         H = SQRT(-1)*OMEGA*I - SYSTEM MATRIX

    GO   COMPLEX MATRIX        3 ROWS        3 COLUMNS
(-2.3652E-03,-5.6223E-10) (  .0000E+00,  .0000E+00) ( 1.3508E-07, 6.8266E-11)
(  .0000E+00,  .0000E+00) (-2.3654E-03,-6.8843E-10) (  .0000E+00,  .0000E+00)
( 1.3508E-07, 6.8266E-11) (  .0000E+00,  .0000E+00) (-3.0447E-03,-1.2059E-08)

KO CALCULATION

 RCOND =     6.4820193E-03
 RCOND = ESTIMATE OF RECIPROCAL OF CONDITION NUMBER FOR
         H = SQRT(-1)*OMEGA*I - SYSTEM MATRIX

    KO   COMPLEX MATRIX        3 ROWS        3 COLUMNS
( 5.8832E+04, 6.5483E+03) ( 1.0497E-03, 6.3063E-05) ( 4.9397E+02, 1.1730E+02)
( 6.4537E-04,-6.1292E-06) ( 5.8002E+04, 6.4479E+03) (-1.7400E-02,-2.2371E-03)
( 4.7052E+02, 1.1765E+02) (-3.8365E-03,-4.3116E-04) ( 1.0462E+05, 1.9395E+04)

  G    COMPLEX MATRIX        3 ROWS        3 COLUMNS
(-1.3915E+02,-1.5488E+01) (-2.4834E-06,-1.4922E-07) (-1.1542E+00,-2.7482E-01)
(-1.5265E-06, 1.4497E-08) (-1.3720E+02,-1.5252E+01) ( 4.1157E-05, 5.2914E-06)
(-1.4247E+00,-3.5732E-01) ( 1.1681E-05, 1.3128E-06) (-3.1854E+02,-5.9053E+01)
```

```
 VECTOR SIGMA OF COMPUTED SINGULAR VALUES OF G

 SIGMA(   1) =     .3239745E+03
 SIGMA(   2) =     .1400017E+03
 SIGMA(   3) =     .1380409E+03
```

```
DEMONSTRATION OF FREQ, BASED ON EXAMPLE IN NASA TP-2560
                                                                  FREQ     PROGRAM

PRINTOUT HAS BEEN TURNED OFF FOR THE NEXT 6 ANALYSES.
     THESE ARE THE FEEDBACK TRANSFER MATRIX FOR THE DESIGN MODEL WITH
     COMPENSATOR; AND ALL 5 ANALYSES OFFERED BY "FREQ" USING THE FULL
     26TH-ORDER PLANT WITH THE COMPENSATOR
```

References

Armstrong, Ernest S. 1978a: *ORACLS—A System for Linear-Quadratic-Gaussian Control Law Design.* NASA TP-1106, 1978.

Armstrong, Ernest S. 1978b: *Optimal Regulator Algorithms for Linear-Quadratic Controllers and Optimal Filters.* Tech Brief LAR-12313, Fall.

Armstrong, Ernest S. c.1980: *ORACLS—A Design System for Linear Multivariable Control.* Marcel Dekker, Inc.

Armstrong, E. S.; Joshi, S. M.; and Stewart, E. J. c.1988: Robust Model-Based Controller Synthesis for the SCOLE Configuration. *Proceedings—The Twentieth Southeastern Symposium on System Theory,* IEEE Catalog No. 88CH2553-6, Computer Soc., pp. 646-651.

COSMIC Software Catalog, 1989 ed. NASA CR-183281.

Dongarra, J. J.; Moler, C. G.; Bunch, J. R.; and Stewart, G. W. 1979: *LINPACK Users' Guide.* SIAM.

Dorato, Peter 1987: A Historical Review of Robust Control. *IEEE Control Syst.,* vol. 7, no. 2, Apr., pp. 44-47.

Doyle, John C.; and Stein, Gunter 1981: Multivariable Feedback Design: Concepts for a Classical/Modern Synthesis. *IEEE Trans. Autom. Control,* vol. AC-26, no. 1, Feb., pp. 4-16.

Doyle, John C.; Wall, Joseph E.; and Stein, Gunter c.1982: Performance and Robustness Analysis for Structured Uncertainty. *Proceedings of the 21st IEEE Conference on Decision & Control, Volume 2,* IEEE Catalog No. 82CH1788-9, IEEE Control Systems Soc., pp. 629-636.

Golub, Gene H.; and Van Loan, Charles F. c.1983: *Matrix Computations.* Johns Hopkins Univ. Press.

Hefner, R. D.; and Mingori, D. L. 1987: Robust Controller Design Using Frequency Domain Constraints. *J. Guid., Control, & Dyn.,* vol. 10, no. 2, Mar.-Apr., pp. 158-165.

Joshi, S. M.; and Armstrong, E. S. 1987: Design of Robust Line-of-Sight Pointing Control System for the SCOLE Configuration. *Proceedings of the 1987 American Control Conference, Volume 2,* IEEE Catalog No. 87CH2378-8, American Automatic Control Council, pp. 1125-1127.

Joshi, Suresh M.; Armstrong, Ernest S.; and Sundararajan, N. 1986: *Application of LQG/LTR Technique to Robust Controller Synthesis for a Large Flexible Space Antenna.* NASA TP-2560.

Joshi, S. M.; Rowell, L. R.; and Armstrong, E. S. c.1988: Robust Controller Synthesis for Large Flexible Space Structures. *Recent Advances in Control of Nonlinear and Distributed Parameter Systems, Robust Control, and Aerospace Control Applications,* J. Bentsman and S. M. Joshi, eds., DSC-Vol. 10, American Soc. of Mechanical Engineers, pp. 147-155.

Kwakernaak, Huibert; and Sivan, Raphael c.1972: *Linear Optimal Control Systems.* John Wiley & Sons, Inc.

Laub, Alan J. c.1981: Efficient Multivariable Frequency Response Computations. *IEEE Trans. Autom. Control,* vol. AC-26, no. 2, Apr., pp. 407-408.

Laub, Alan J. 1986: Algorithm 640—Efficient Calculation of Frequency Response Matrices From State Space Models. *ACM Trans. Math. Softw.,* vol. 12, no. 1, Mar., pp. 26-33.

Lehtomaki, Norman A. 1981: *Functional Description of Computer Programs for the Robustness Analysis of Multivariable Control Systems.* LIDS-R-1094 (Contract DE-AC01-78RA03395), Massachusetts Inst. of Technology, June.

Mukhopadhyay, V. 1987: Stability Robustness Improvement Using Constrained Optimization Techniques. *J. Guid., Control, & Dyn.,* vol. 10, no. 2, Mar.-Apr., pp. 172-177.

Multi-Variable Analysis and Design Techniques 1981. AGARD-LS-117, Sept.

Oppenheim, Alan V.; Willsky, Alan S.; and Young, Ian T. c.1983: *Signals and Systems.* Prentice-Hall, Inc.

Ridgely, D. Brett; and Banda, Siva S. 1986: *Introduction to Robust Multivariable Control.* AFWAL-TR-85-3102, U.S. Air Force, Feb. (Available from DTIC as AD A165 891.)

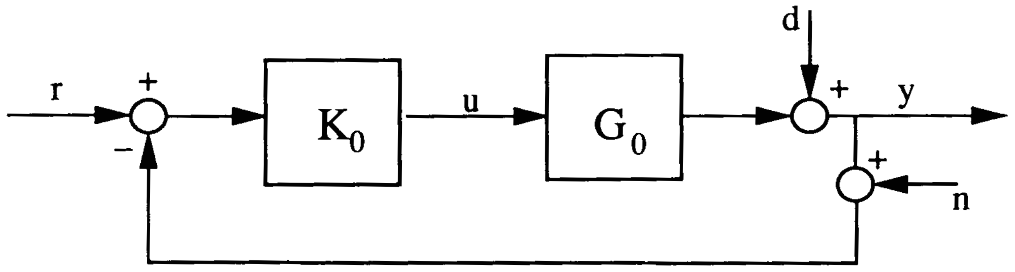
Safonov, Michael G.; Laub, Alan J.; and Hartmann, Gary L. 1981: Feedback Properties of Multivariable Systems: The Role and Use of the Return Difference Matrix. *IEEE Trans. Autom. Control,* vol. AC-26, no. 1, Feb., pp. 47-65.

Smith, B. T.; Boyle, J. M.; Dongarra, J. J.; Garbow, B. S.; Ikebe, Y.; Klema, V. C.; and Moler, C. B. 1976: Matrix Eigensystem Routines—EISPACK Guide, Second ed. *Volume 6 of Lecture Notes in Computer Science,* G. Goos and J. Hartmanis, eds., Springer-Verlag.

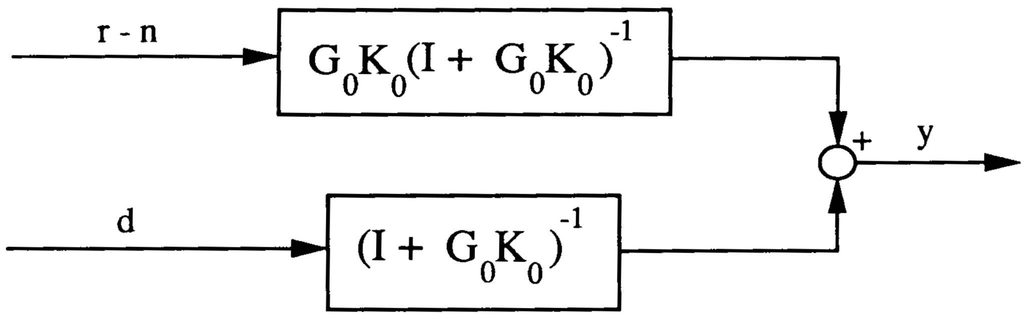
Stein, Gunter; and Athans, Michael 1987: The LQG/LTR Procedure for Multivariable Feedback Control Design. *IEEE Trans. Autom. Control,* vol. AC-32, no. 2, Feb., pp. 105-114.

Sundararajan, N.; Joshi, S. M.; and Armstrong, E. S. 1987: Robust Controller Synthesis for a Large Flexible Space Antenna. *J. Guid., Control, & Dyn.,* vol. 10, no. 2, Mar.-Apr., pp. 201-208.

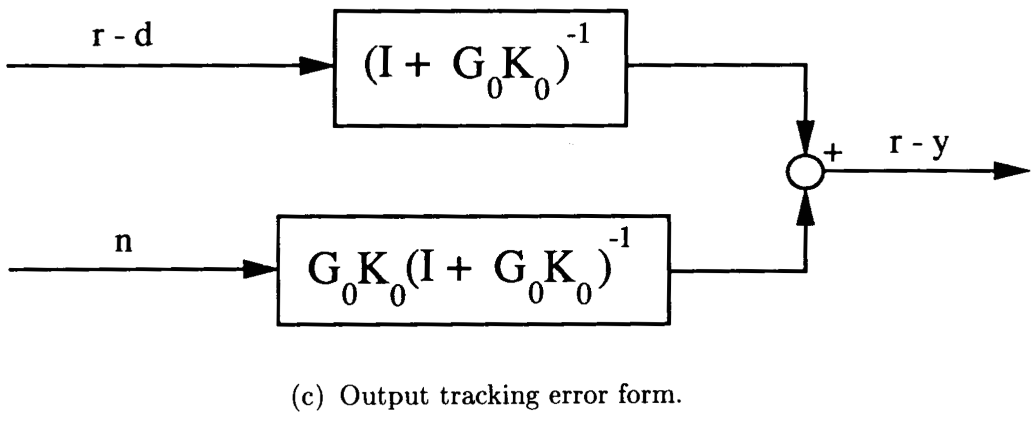
C-2



(a) Unity-gain feedback form.



(b) Output form.



(c) Output tracking error form.

Figure 1. Equivalent forms of the standard feedback configuration.

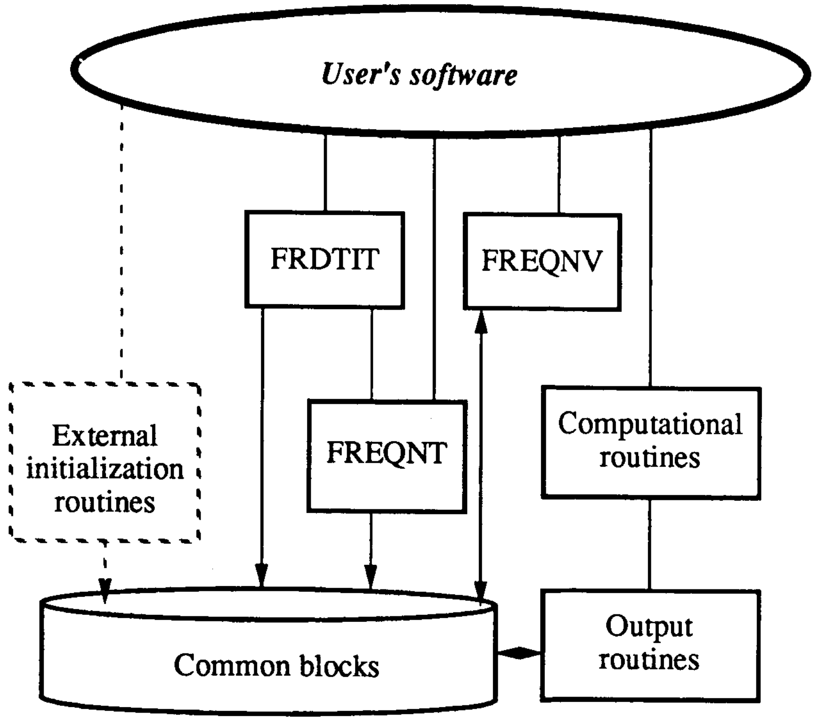


Figure 2. Initialization and output structure of FREQ.

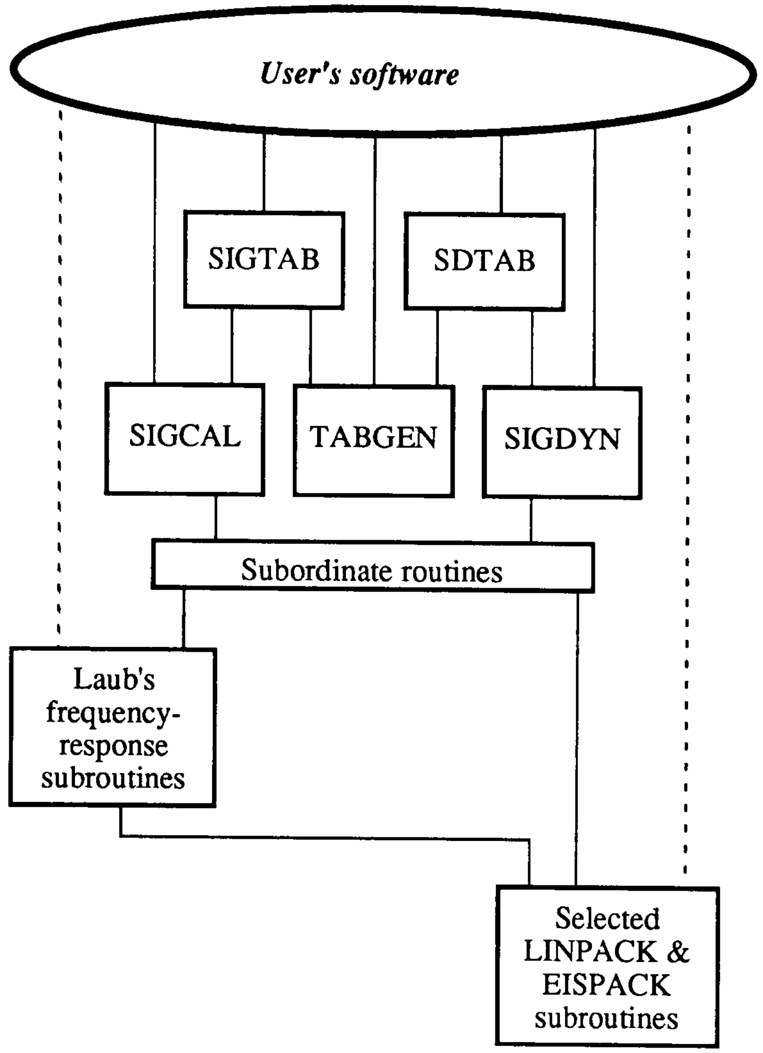

User's software
SIGTAB
SDTAB
SIGCAL
TABGEN
SIGDYN
Subordinate routines
Laub's frequency-response subroutines
Selected LINPACK & EISPACK subroutines

Figure 3. Computational routines structure of FREQ.

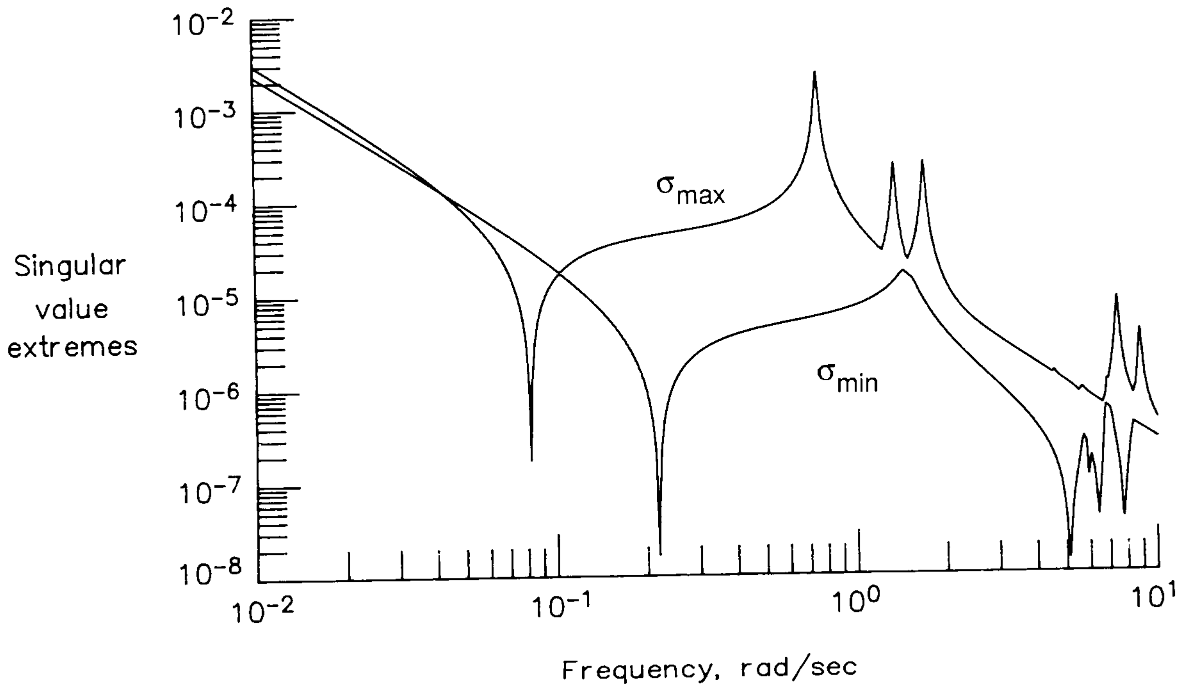


Figure 4. Loop-gain transfer matrix (open-loop response) for the 26th-order model of the hoop/column antenna (**G**).

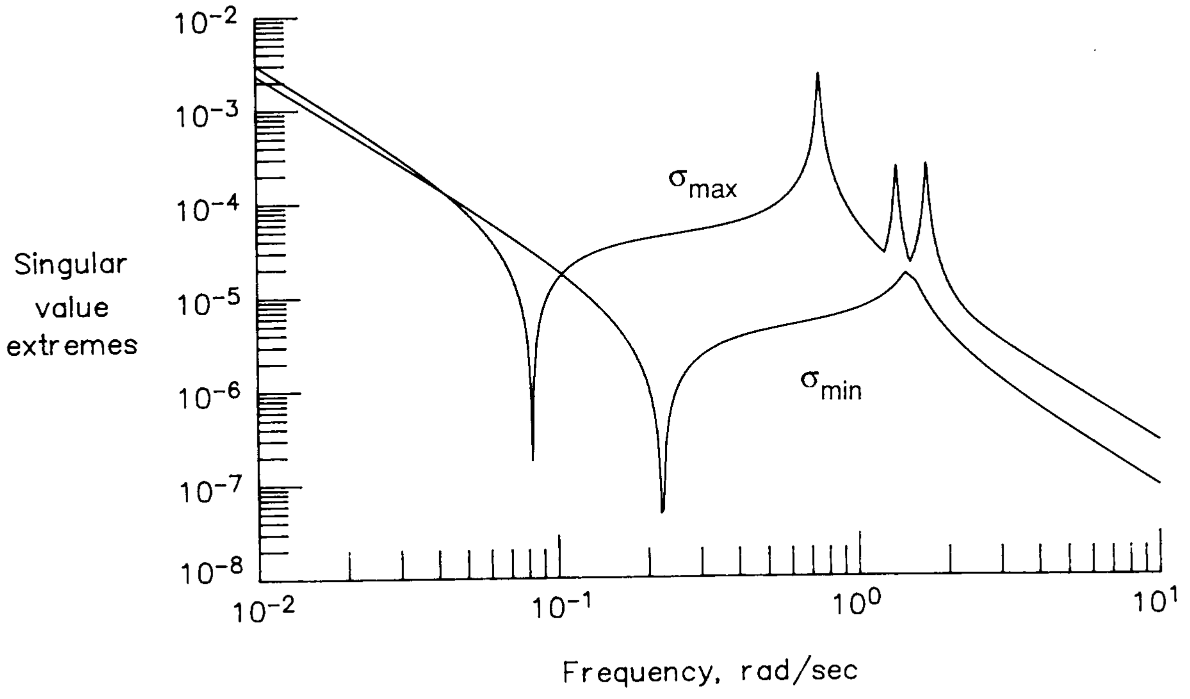


Figure 5. Loop-gain transfer matrix (open-loop response) for the 12th-order design model ($\mathbf{G}_p$).

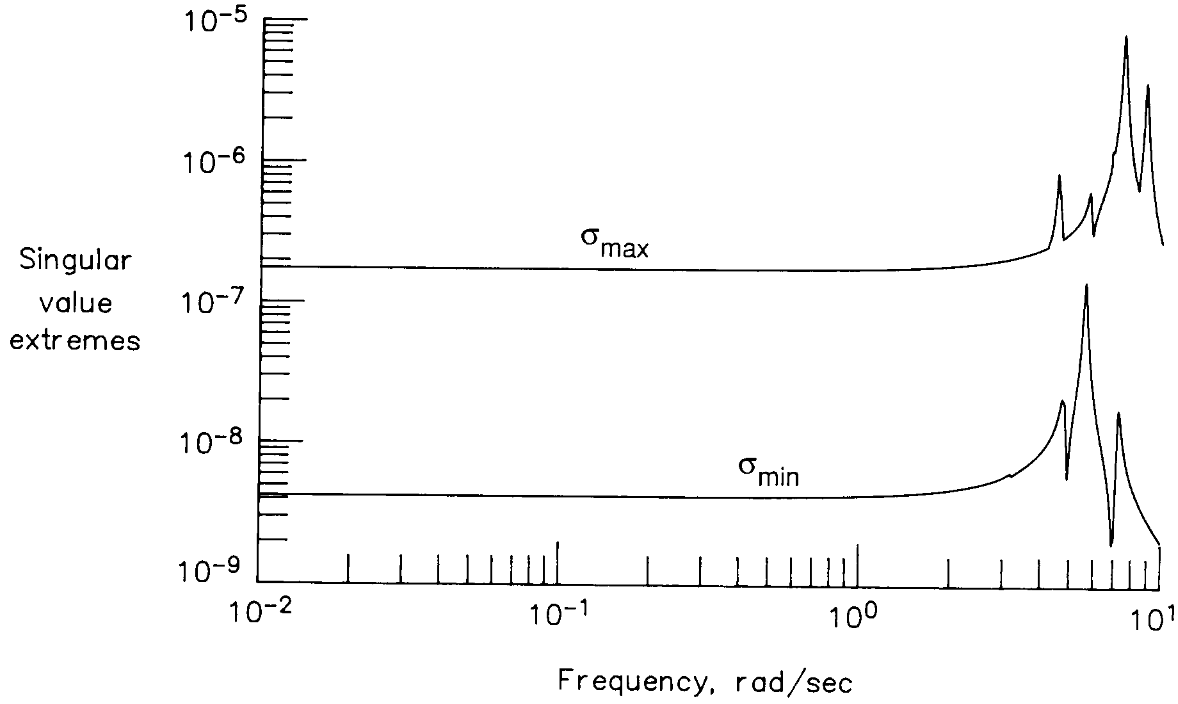


Figure 6. Loop-gain transfer matrix (open-loop response) for the additive-error model ($\Delta\mathbf{G}$).

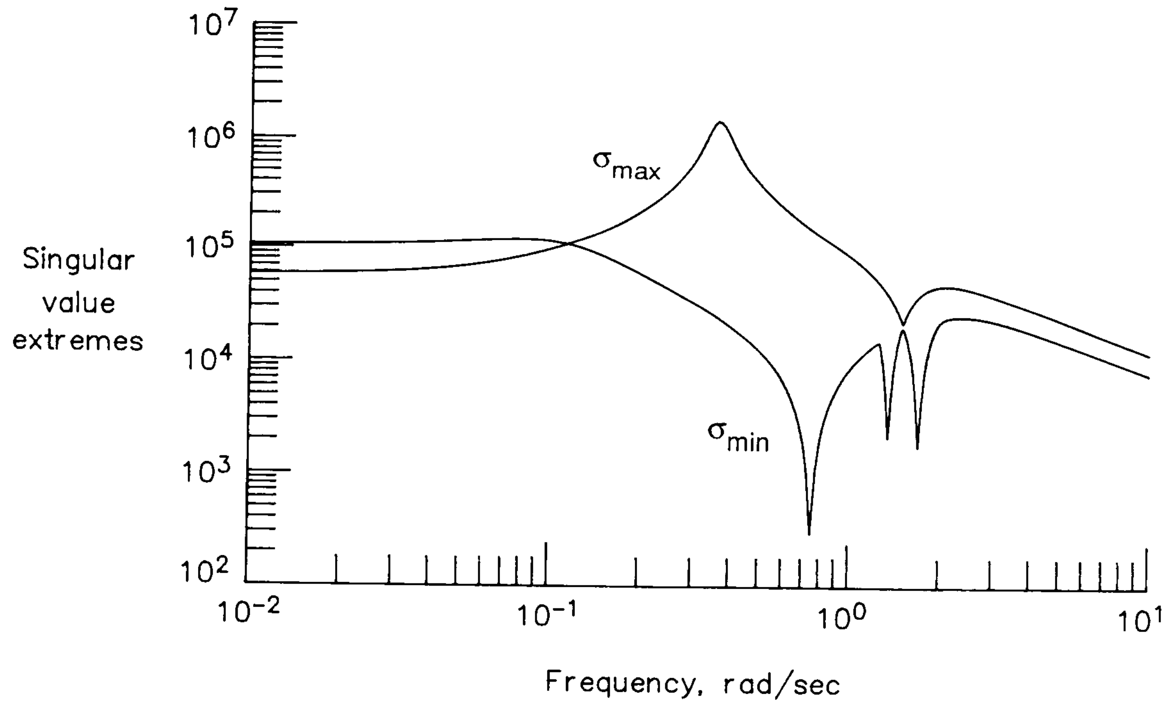


Figure 7. Loop-gain transfer matrix (open-loop response) for the compensator ($\mathbf{G}_c$).

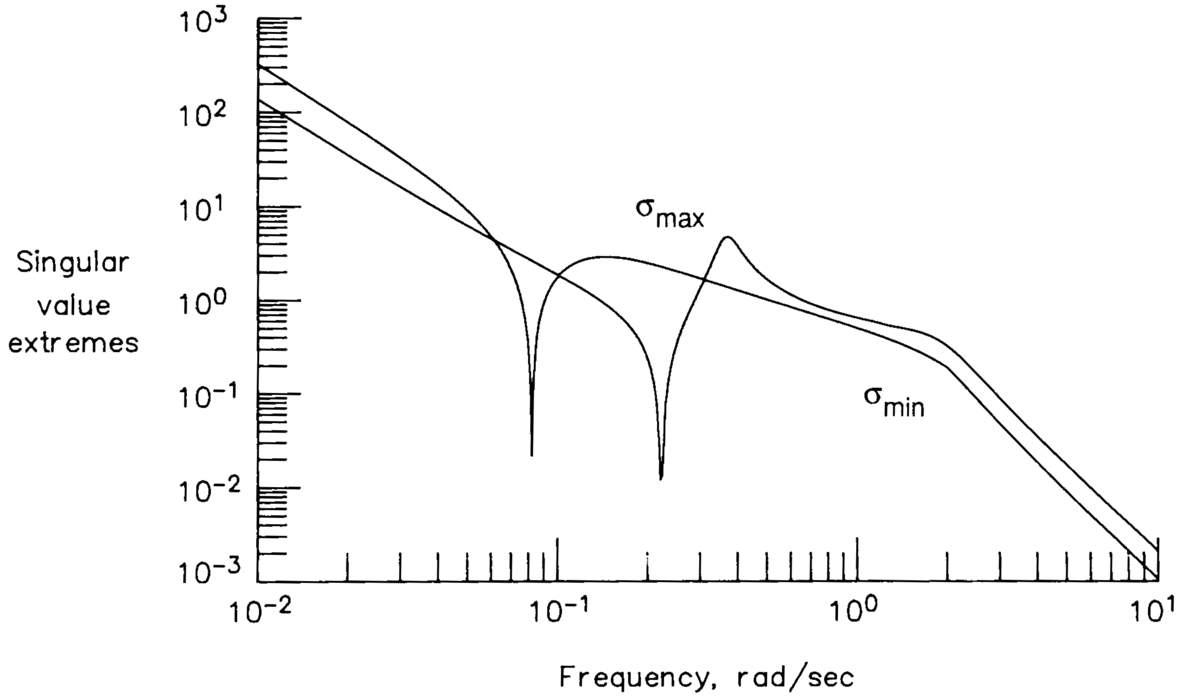


Figure 8. Loop-gain transfer matrix (open-loop response) for the design model with compensator ($\mathbf{G}_p\mathbf{G}_c$).

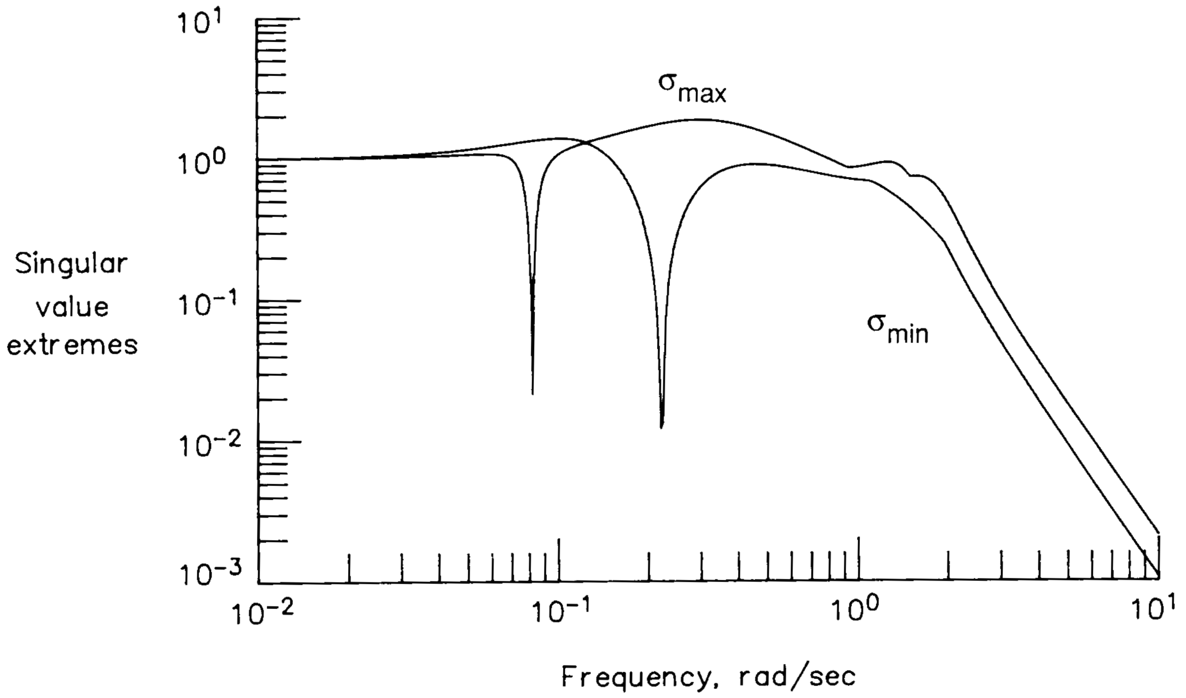


Figure 9. Feedback transfer matrix (closed-loop response) for the design model with compensator ($\mathbf{G}_p\mathbf{G}_c(\mathbf{I}+\mathbf{G}_p\mathbf{G}_c)^{-1}$).

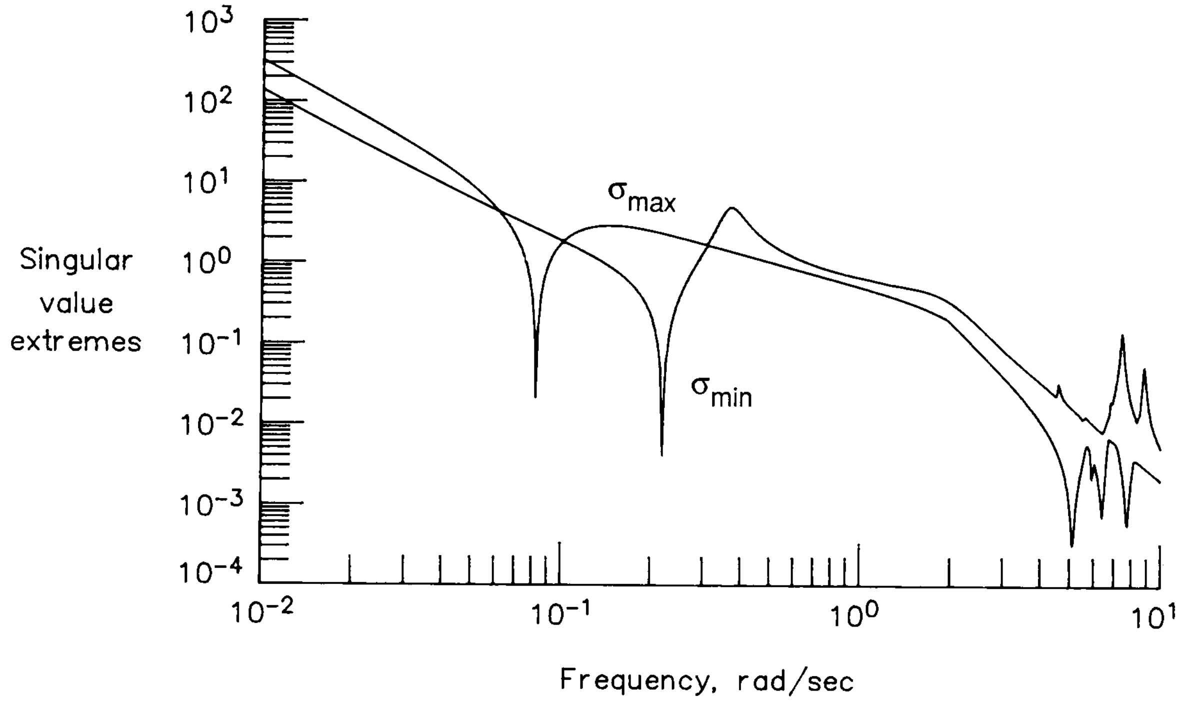


Figure 10. Loop-gain transfer matrix (open-loop response) for the hoop/column antenna with compensator ($\mathbf{GG}_c$).

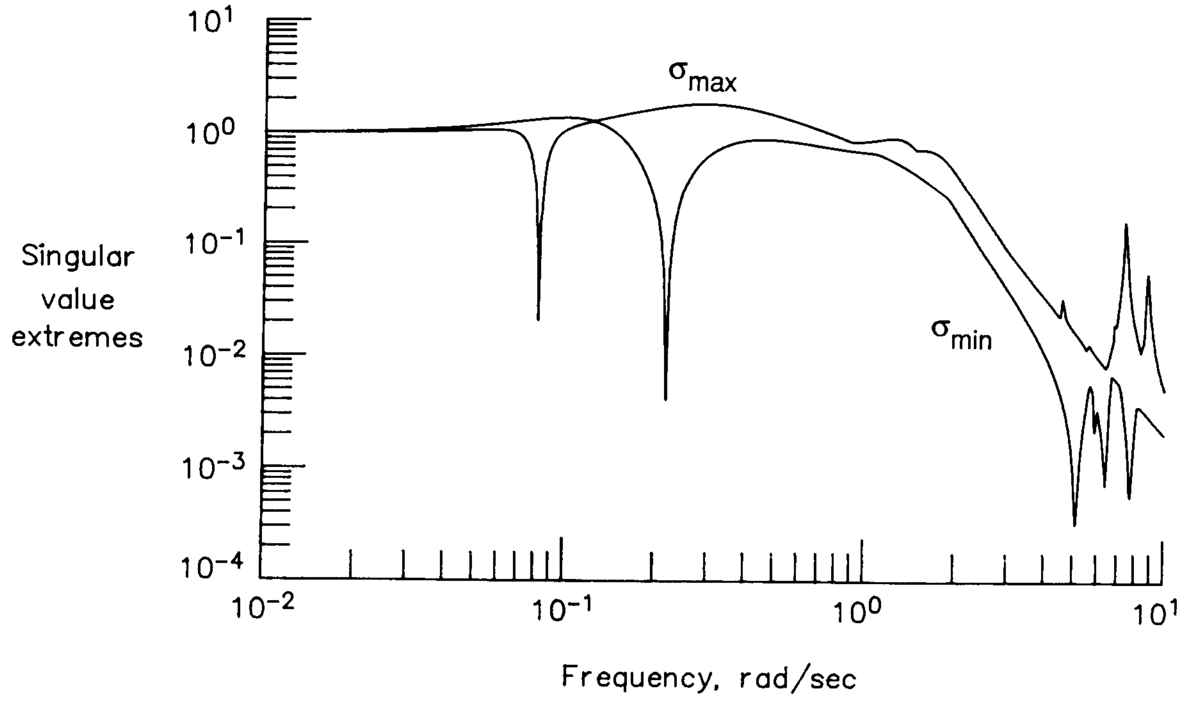


Figure 11. Feedback transfer matrix (closed-loop response) for the hoop/column antenna with compensator ($\mathbf{GG}_c(\mathbf{I}+\mathbf{GG}_c)^{-1}$).

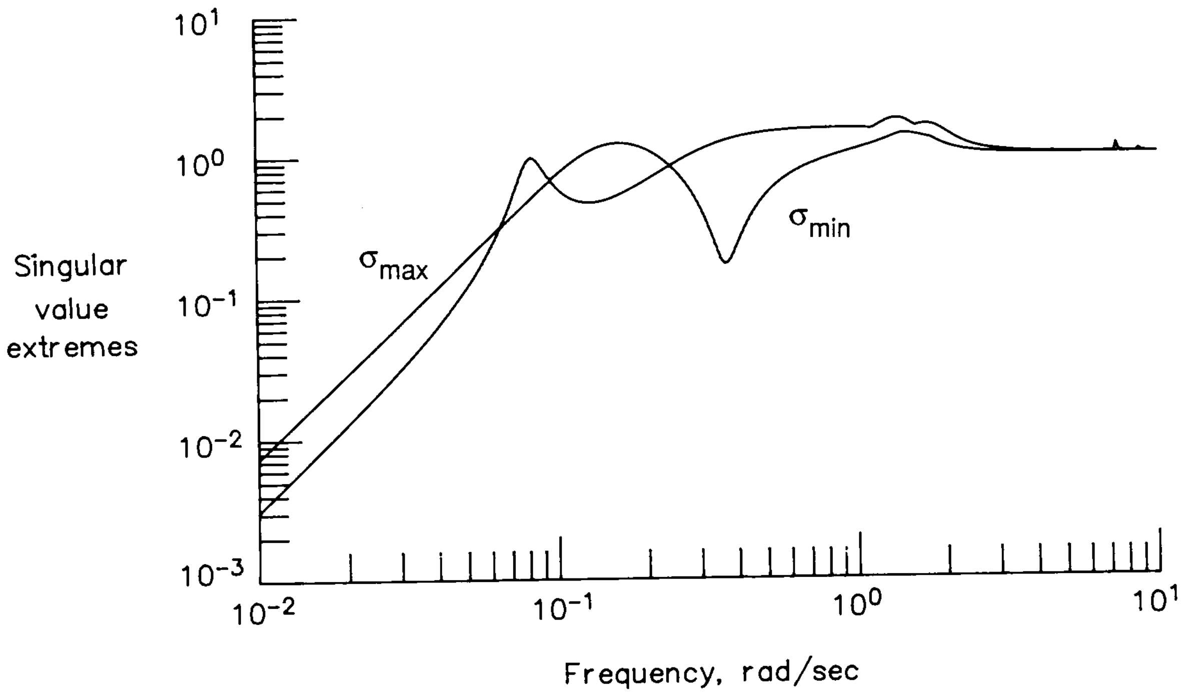


Figure 12. Sensitivity matrix for the hoop/column antenna with compensator $((\mathbf{I} + \mathbf{G}\mathbf{G}_c)^{-1})$.

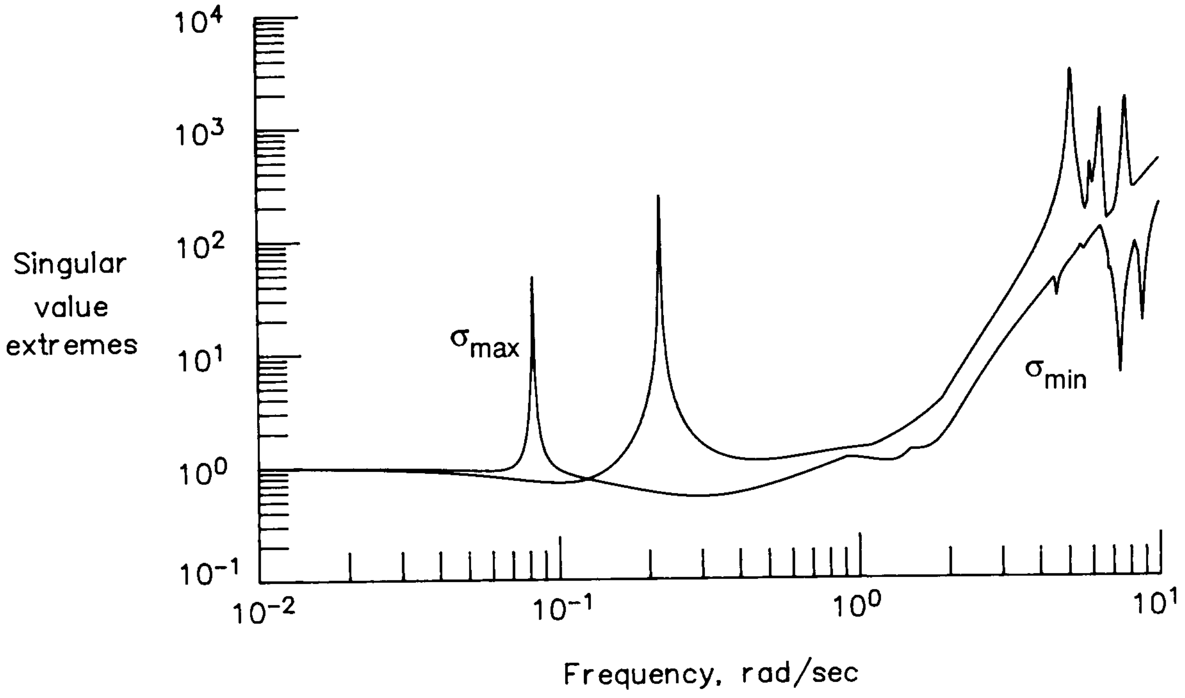


Figure 13. Inverse return difference matrix for the hoop/column antenna with compensator $(\mathbf{I} + (\mathbf{G}\mathbf{G}_c)^{-1})$

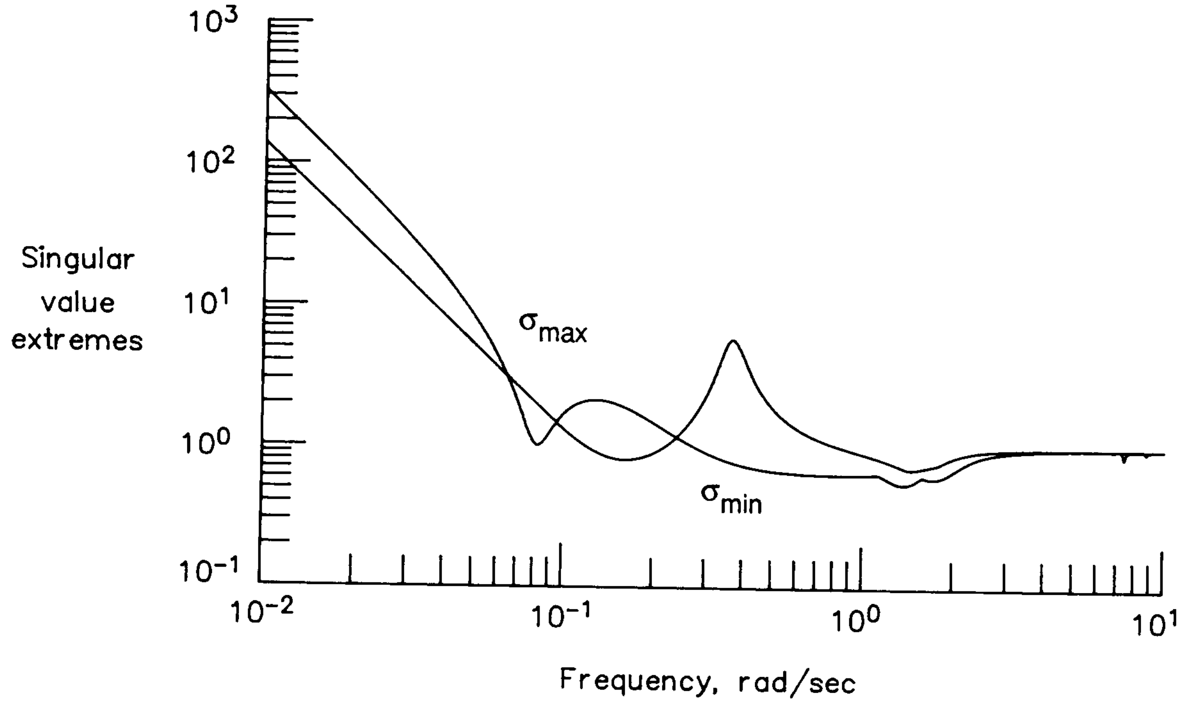


Figure 14. Return difference matrix for the hoop/column antenna with compensator $(\mathbf{I} + \mathbf{GG}_c)$.

National Aeronautics and Space Administration

Report Documentation Page

1. Report No. NASA TM-4127	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle FREQ: A Computational Package for Multivariable System Loop-Shaping Procedures		5. Report Date September 1989
		6. Performing Organization Code
7. Author(s) Daniel P. Giesy and Ernest S. Armstrong		8. Performing Organization Report No. L-16566
9. Performing Organization Name and Address NASA Langley Research Center Hampton, VA 23665-5225		10. Work Unit No. 506-46-11-01
		11. Contract or Grant No.
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, DC 20546-0001		13. Type of Report and Period Covered Technical Memorandum
		14. Sponsoring Agency Code

15. Supplementary Notes
Daniel P. Giesy: PRC Kentron, Inc., Aerospace Technologies Division, Hampton, Virginia.
Ernest S. Armstrong: Langley Research Center, Hampton, Virginia.

16. Abstract
Many approaches in the field of linear, multivariable time-invariant systems analysis and controller synthesis employ loop-shaping procedures wherein design parameters are chosen to shape frequency-response singular value plots of selected transfer matrices. This report documents a software package, FREQ, for computing within one unified framework many of the most used multivariable transfer matrices for both continuous and discrete systems. The matrices are evaluated at user-selected frequency-response values, and singular values and vectors are computed. FREQ also tabulates maximum and minimum singular values against frequency. Example computations are presented to demonstrate the use of the FREQ code.

17. Key Words (Suggested by Authors(s)) Software package; Loop shaping; Multivariable frequency response; Singular values and vectors; Sigma plots	18. Distribution Statement Unclassified—Unlimited; Subject Category 61

19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 106	22. Price A06

NASA FORM 1626 OCT 86 NASA-Langley, 1989

For sale by the National Technical Information Service, Springfield, Virginia 22161-2171